RxGS: Receiver-Generalizable 3D Gaussian Splatting for Radio-Frequency Data Synthesis

Kang Yang **Mani Srivastava**
Department of Electrical and Computer Engineering
University of California, Los Angeles
kyang73@g.ucla.edu mbs@ucla.edu

Abstract

Radio-frequency (RF) data synthesis predicts the received signal given transmitter and receiver positions, and is essential for wireless applications. Recent 3D Gaussian Splatting (3DGS)-based methods achieve efficient synthesis at any transmitter location within a surveyed scene, but require a separate model for each receiver. We therefore present RxGS, a shared 3DGS model that first learns receiver-independent Gaussian geometry and then freezes it while learning directional radiance conditioned on receiver position. Its global conditioning branch captures receiver-dependent effects shared across the scene, while a local branch models per-scatterer variations from the receiver's geometry and occlusion. The shared geometry also supports batched rendering through a multi-receiver rasterizer. Across RF datasets, RxGS predicts at receiver positions excluded from training within the surveyed scene while significantly reducing training, inference, and storage costs. Project page: https://kangyangg.com/projects/rxgs/.

1 Introduction

Radio-frequency (RF) data, such as received signal strength indicator (RSSI) and channel state information (CSI), quantifies how RF signals propagate from a transmitter to a receiver in a scene [1]. Such data is essential for wireless networking and sensing applications [2–8]. For example, network planning [3] requires coverage maps that evaluate signal strength across candidate transmitter and receiver placements. Fingerprint-based localization [5, 8, 9] depends on databases of RF measurements collected across numerous transmitter positions at multiple receivers. However, acquiring these measurements requires site surveys at thousands of transmitter–receiver positions, which is prohibitively labor-intensive and time-consuming [10–13].

RF data synthesis reduces the need for exhaustive site surveys. Classical ray tracing simulations [14–17] trace propagation paths through Computer-Aided Design (CAD) scene models. However, accurate CAD models require precise scene geometry, material permittivity, and conductivity. These properties are difficult to obtain in practice, making simulations infeasible.

Neural scene representations sidestep these requirements by learning RF scene responses from measurements [5, 18–21]. This enables synthesis of novel RF data via differentiable rendering, without precise CAD models or material priors. Neural Radiance Field (NeRF)-based RF methods [5, 21] pioneer this direction but incur prohibitive computational costs from dense volumetric sampling and multi-layer perceptron (MLP) inference. 3D Gaussian Splatting (3DGS)-based RF methods [18–20] replace volumetric sampling with 3D Gaussians for faster training and inference.

The rendering interface of 3DGS also shapes how these methods represent the two endpoints of an RF link. Optical 3DGS [22–24] is parameterized by a single variable, the camera viewpoint, since the scene is queried from one viewpoint at a time. RF data synthesis instead depends on two variables, the transmitter position and the receiver position. Existing 3DGS-based RF methods expose the densely surveyed moving endpoint as an input and absorb the other, fixed endpoint into model weights. In our data, the moving transmitters are surveyed across thousands of positions, while the receivers occupy a

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

smaller set of fixed positions. Without an input pathway for the fixed receiver position, these models learn receiver-specific radiance coefficients, as detailed in §2.

However, multi-receiver deployments in a scene are the norm in practice [11, 25, 26]. For example, a Bluetooth Low Energy (BLE) localization system deploys dozens of gateways distributed across a facility [5]. With a separate checkpoint for each receiver, the costs of a multi-receiver deployment grow in two ways. ❶ *Scaling Cost*: synthesizing RF data for N receivers within one scene requires training N independent models. ❷ *Adding New Receivers.* For per-receiver 3DGS methods, a new fixed receiver in the same scene requires a full transmitter survey and its own checkpoint.

We present RxGS, a 3DGS-based RF data synthesis framework that conditions one shared model on fixed receiver locations within a surveyed scene. Our key insight is that scene geometry is receiver-independent while directional radiance is not: scatterers such as walls and furniture remain fixed regardless of receiver position, but the angles and distances to the receiver change. This motivates a two-stage training procedure. Stage I learns shared 3D Gaussian geometry of the scene. Stage II freezes this geometry and jointly learns receiver-conditioned directional radiance.

When the receiver moves, both the global radiance pattern and each Gaussian's individual contribution change. To capture these effects, we propose two complementary conditioning modules. A global module modulates harmonic coefficients based on receiver position, capturing the shared component of receiver-dependent variation. A local module adjusts each Gaussian based on its direction, distance, and occlusion relative to the receiver, capturing per-scatterer residuals. Both modules are radiance-representation-agnostic and apply to existing 3DGS-based RF methods, including spherical harmonics (SH), Fourier–Legendre expansions (FLE), and deformation networks [18–20]. For the SH and FLE backbones, a batched rasterizer uses the Compute Unified Device Architecture (CUDA) to render N receiver outputs in parallel while sharing projection and sorting.

RxGS introduces explicit fixed-receiver conditioning to 3DGS-based RF synthesis, enabling predictions at receiver locations excluded from training within the surveyed region. On RSSI, it improves the RF-3DGS [18] and GSRF [20] per-receiver baselines. With the WRF-GS+ [19] backbone, it eliminates two high-error outliers with a 0.33 dBm increase in mean absolute error (MAE). On spatial spectrum and CSI, it trails per-receiver baselines by 0.8 to 1.7 dB. Training is up to $45\times$ faster, batched FLE/SH inference up to $7.6\times$ faster, and storage up to $24\times$ smaller.

2 Preliminaries

We review the 3DGS-based RF data synthesis framework [18–20] that RxGS builds on.

1) Scene Representation. The RF scene, *e.g.*, a conference room, is a collection of 3D Gaussian primitives $\{\mathcal{G}_k\}_{k=1}^{K}$, each defined by four attributes:

$$\mathcal{G}_k = (\mathbf{p}_k, \mathbf{C}_k, \phi_k, \tau_k). \tag{1}$$

The mean $\mathbf{p}_k \in \mathbb{R}^3$ and covariance $\mathbf{C}_k \in \mathbb{R}^{3\times3}$ define the spatial geometry of each Gaussian. The radiance ϕ_k represents the RF signal emitted by the k-th Gaussian. To model its directional dependency, ϕ_k is parameterized by the coefficients of basis functions on the sphere, such as SH [27] or FLE [28]. Transmittance τ_k models direction-independent attenuation through the k-th Gaussian.

2) Rendering. Given a receiver at position $\mathbf{r}$ and a transmitter at position $\mathbf{t}$, RF data is synthesized as follows. Rays are emitted from an emission surface centered at the transmitter $\mathbf{t}$. Each Gaussian is projected onto this surface to identify ray–Gaussian intersections based on the ray direction and the Gaussian mean $\mathbf{p}_k$ and covariance $\mathbf{C}_k$. For each intersecting Gaussian along a ray, the directional radiance ϕ_k depends on the direction from the transmitter to the Gaussian center:

$$\hat{\mathbf{d}}_k = (\mathbf{p}_k - \mathbf{t}) \;/\; \|\mathbf{p}_k - \mathbf{t}\|_2. \tag{2}$$

The received signal for each ray aggregates the radiance ϕ_k and transmittance τ_k contributions of all intersecting Gaussians along that ray. All Gaussian attributes are then jointly optimized by minimizing a reconstruction loss between rendered and ground-truth RF measurements.

Receiver Dependence. In our setting, RF data synthesis involves a densely surveyed moving endpoint and a fixed receiver endpoint. In our datasets, the moving endpoint is the transmitter at $\mathbf{t}$ and the fixed endpoint is the receiver at $\mathbf{r}$. Across prior formulations, the physical transmitter and receiver labels can be reversed. With matched hardware, reciprocity aligns their fixed and moving endpoint roles, while directional measurements retain endpoint-specific conventions. In Equation 2, the moving

transmitter enters explicitly through $\hat{\mathbf{d}}_k$, enabling synthesis at new transmitter positions. The fixed receiver position $\mathbf{r}$ enters implicitly through the ground-truth measurements used during training. Consequently, each checkpoint represents the radiance field for its training receiver. Additional receiver placements call for separate measurements and fitting.

3 Related Work

NeRF-Based RF Data Synthesis. Optical neural rendering methods [29–32] have demonstrated strong capabilities in synthesizing visual content. This paradigm has also been extended to the RF domain. NeRF^2 [5] pioneers neural RF data synthesis by representing the scene as a continuous volumetric function parameterized by two large MLPs for attenuation and radiance. Several subsequent works study neural RF synthesis [21, 33–36]. For example, NeWRF [21] leverages direction-of-arrival (DoA) priors to reduce the number of emitted rays, improving computational efficiency. GRaF [33] interpolates spatial spectra from geographically proximate transmitters, but requires reference spectra at the query receiver at inference, preventing zero-shot prediction at unseen receivers. NeRF-based methods such as NeRF^2 represent receiver position through the ray origin and sampled directions, allowing queries at previously unseen receivers. Dense volumetric sampling and repeated MLP queries raise training and inference costs [29, 30].

3DGS-Based RF Data Synthesis. Existing 3DGS-based RF methods [18–20, 37, 38] fall into two categories based on whether neural networks are adopted. WRF-GS+ [19] uses a deformation network to predict offsets for Gaussian attributes from Gaussian and transmitter positions, and SwiftWRF [37] follows a similar architecture with 2D Gaussians for faster rendering. RF-3DGS [18] and GSRF [20] avoid neural networks. RF-3DGS learns geometry from camera images, then retrains SH coefficients from RF measurements. GSRF optimizes per-Gaussian FLE radiance coefficients [28] as learnable complex-valued parameters. These methods supply the moving endpoint as a spatial query and encode the fixed endpoint in a checkpoint, as described in §2. A multi-receiver deployment therefore trains one checkpoint per fixed endpoint. WRF-GS+ labels that fixed endpoint as the receiver, while RF-3DGS and GSRF label it as the transmitter.

Radio Maps from Scene Layouts. RadioUNet [39] predicts scalar pathloss maps from 2D occupancy maps and transmitter positions using simulated data. By contrast, RxGS synthesizes RSSI, spatial spectra, and CSI from RF measurements, including floorplan-free BLE data.

Relighting in Neural Rendering. Generalizing across receiver positions in RF synthesis is structurally analogous to relighting in optical neural rendering, where both must generalize across a variable baked into the learned representation. Several works address this by replacing the standard rendering equation with a physically-based rendering (PBR) equation that takes lighting as an explicit input [40–44]. For instance, Relightable 3DGS [42] augments each Gaussian with surface normals, bidirectional reflectance distribution function (BRDF) parameters, and incident light to compute outgoing radiance under arbitrary illumination. Optical relighting uses BRDF-based priors, whereas RF multipath calls for a receiver-dependent signal model tailored to propagation. RxGS therefore introduces lightweight learned conditioning modules to capture these receiver-dependent effects. Its global module plays a role analogous to a learned environment map, while the global-only and additive variants in Table 3 isolate the effects of the two conditioning choices.

4 Method

Consider a scene with transmitter positions $\{\mathbf{t}_i\}_{i=1}^{M}$ and receiver positions $\{\mathbf{r}_j\}_{j=1}^{N}$. For each transmitter–receiver pair $(\mathbf{t}_i, \mathbf{r}_j)$, the received RF data $S(\mathbf{t}_i, \mathbf{r}_j)$ is measured. S takes different forms across modalities: a scalar RSSI value, a spatial spectrum matrix, or complex-valued CSI.

Prior Formulation (Single-Receiver). Existing 3DGS-based RF data synthesis methods learn a separate model Θ_j for each receiver $\mathbf{r}_j$:

$$\hat{S}(\mathbf{t}, \mathbf{r}_j) = f(\mathbf{t};\, \Theta_j), \quad \forall\, \mathbf{t} \in \mathbb{R}^3. \tag{3}$$

The per-receiver formulation requires N checkpoints for N receivers. Extending the deployment to $\mathbf{r}^* \notin \{\mathbf{r}_j\}$ requires receiver-specific measurements and model training.

Our Formulation (Receiver-Generalizable). Our goal is to learn a single model Θ that generalizes across transmitter positions and fixed receiver positions within a surveyed, static scene:

$$\hat{S}(\mathbf{t}, \mathbf{r}) = f(\mathbf{t}, \mathbf{r};\, \Theta), \quad \mathbf{t} \in \mathcal{T}_{\text{survey}},\, \mathbf{r} \in \mathcal{R}_{\text{survey}}, \tag{4}$$

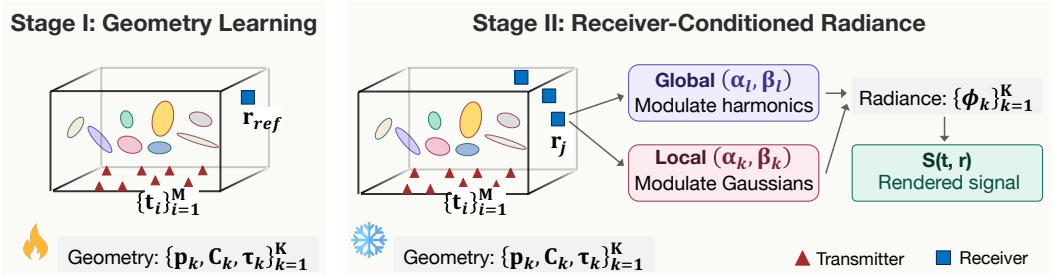


Figure 1: RxGS's two-stage architecture. Stage I learns receiver-independent 3D Gaussian geometry, and Stage II conditions radiance on the receiver via global and local modules in one unified model.

Here $\mathcal{T}_{\text{survey}}$ and $\mathcal{R}_{\text{survey}}$ denote transmitter and candidate receiver locations in the surveyed deployment region, including receiver sites excluded from training. The key challenge is to capture receiver-dependent variation in one parameter set while managing the accuracy–efficiency tradeoff.

4.1 Geometry-Radiance Decomposition Workflow

The key insight behind RxGS is that the 3D Gaussian attributes partition into two groups with distinct dependencies. The geometric attributes, position $\mathbf{p}_k$, covariance $\mathbf{C}_k$, and transmittance τ_k, describe where scatterers are located, their spatial extent, and how they attenuate signals. These are physical properties of the environment and are independent of the receiver position. The radiance coefficients ϕ_k, in contrast, encode the directional signal response at each Gaussian and change with receiver position. Appendix A provides a theoretical analysis showing that, under the rendering abstraction adopted by 3DGS-based RF methods, receiver-dependent variation is naturally expressed through the radiance term while the geometric attributes remain receiver-independent. Within this rendering abstraction, receiver-dependent effects are assigned to radiance while shared geometry supports all receivers. This motivates the two-stage training procedure illustrated in Figure 1.

Stage I: Geometry. All Gaussian attributes $(\mathbf{p}_k, \mathbf{C}_k, \phi_k, \tau_k)$ are jointly optimized on RF measurements from a single reference receiver $\mathbf{r}_{\text{ref}}$ to learn the shared scene geometry. After convergence, we freeze $\{\mathbf{p}_k, \mathbf{C}_k, \tau_k\}_{k=1}^{K}$ for Stage II and discard the Stage I radiance coefficients.

Stage II: Receiver-Conditioned Radiance. With the geometry fixed, we introduce new radiance coefficients conditioned on the receiver position $\mathbf{r}$:

$$\phi_k(\hat{\mathbf{d}}_k, \mathbf{r}) = g(\phi_k^{\text{base}}, \hat{\mathbf{d}}_k, \mathbf{r}; \Theta_{\text{cond}}), \tag{5}$$

where ϕ_k^{base} are learnable base coefficients shared across all receivers, and g is a conditioning function that modulates them based on $\mathbf{r}$. g consists of a global module that modulates directional harmonics and a local module that modulates individual Gaussians, together parameterized by Θ_{cond}. Stage II is trained on RF measurements from all available receivers $\{\mathbf{r}_j\}_{j=1}^{N}$ jointly, allowing the conditioning function to learn how receiver position affects directional radiance.

4.2 Stage I: Geometry Learning

Objective. We select an arbitrary reference receiver $\mathbf{r}_{\text{ref}}$ and optimize all Gaussian attributes $(\mathbf{p}_k, \mathbf{C}_k, \phi_k, \tau_k)$ jointly using RF measurements from all transmitter positions. The training objective minimizes the prediction error between the rendered and measured RF data:

$$\{\mathbf{p}_k, \mathbf{C}_k, \phi_k, \tau_k\}_{k=1}^{K} = \underset{\{\mathbf{p}_k, \mathbf{C}_k, \phi_k, \tau_k\}}{\arg\min} \frac{1}{M} \sum_{i=1}^{M} \ell\Big(\hat{S}(\mathbf{t}_i),\ S(\mathbf{t}_i, \mathbf{r}_{\text{ref}})\Big), \tag{6}$$

where ℓ is a task-specific loss determined by the RF data modality: ℓ_1 for scalar RSSI, ℓ_2 for complex-valued CSI, or a composite loss for spatial spectrum data. The rendered signal $\hat{S}(\mathbf{t}_i)$ is computed via the rendering equation in Equation (21) of Appendix A. Because Gaussian geometry is receiver-independent, this objective learns shared attributes from measurements at the reference receiver.

Representation Agnosticism. Stage I is compatible with any directional radiance representation, including spherical harmonics [27], MLP-based radiance networks [19], or Fourier–Legendre expansions [28], since the decomposition is independent of how ϕ_k is parameterized. After convergence, we freeze $\{\mathbf{p}_k, \mathbf{C}_k, \tau_k\}_{k=1}^{K}$ and discard the Stage I radiance coefficients.

4.3 Stage II: Receiver-Conditioned Radiance

Objective. With the geometry frozen from Stage I, we introduce a conditioning function g_Θ that produces receiver-specific radiance from a shared set of learnable base coefficients $\{\phi_k^{\text{base}}\}_{k=1}^K$:

$$\phi_k(\hat{\mathbf{d}}_k, \mathbf{r}) = g_\Theta(\phi_k^{\text{base}}, \hat{\mathbf{d}}_k, \mathbf{r}). \tag{7}$$

The training objective optimizes $\{\phi_k^{\text{base}}\}_{k=1}^K$ and Θ jointly across all available receivers:

$$\{\phi_k^{\text{base}}, \Theta\} = \underset{\{\phi_k^{\text{base}}, \Theta\}}{\arg\min} \frac{1}{MN} \sum_{j=1}^{N} \sum_{i=1}^{M} \ell\Big(\hat{S}(\mathbf{t}_i, \mathbf{r}_j),\ S(\mathbf{t}_i, \mathbf{r}_j)\Big). \tag{8}$$

The key question is how to design g_Θ. A naive approach would condition on the raw receiver position $\mathbf{r}$ alone, but the decomposition reveals richer structure: the receiver-dependence of ϕ_k arises specifically from the last-segment propagation from each scatterer to the receiver. We formalize this below and derive three constraints that dictate the architecture of g_Θ: the modulation should be affine, it should vary per scatterer, and it can share weights across scatterers.

4.3.1 Design Constraints from the Propagation Factor

Proposition 1 (Last-Segment Factorization)**.** *Under the multipath model in Definition 1 of Appendix A, the radiance of Gaussian k at position $\mathbf{p}_k$ factorizes into a receiver-independent base ϕ_k^{base} and a receiver-dependent last-segment propagation factor η_{eff}:*

$$\phi_k(\hat{\mathbf{d}}_k, \mathbf{r}) = \eta_{\text{eff}}(\mathbf{p}_k, \mathbf{r}) \cdot \phi_k^{\text{base}}(\hat{\mathbf{d}}_k, \mathbf{t}), \tag{9}$$

where ϕ_k^{base} aggregates all path contributions from the transmitter to scatterer k, and:

$$\eta_{\text{eff}}(\mathbf{p}_k, \mathbf{r}) = \underbrace{\frac{\lambda}{4\pi\|\mathbf{p}_k - \mathbf{r}\|} \cdot e^{-j\frac{2\pi}{\lambda}\|\mathbf{p}_k - \mathbf{r}\|}}_{\textit{free-space: path loss + phase}} \cdot \underbrace{\prod_{m \in \mathcal{I}(k,\mathbf{r})} \tau_m}_{\textit{occlusion}} = \frac{\lambda}{4\pi d_k} \cdot e^{-j\frac{2\pi}{\lambda} d_k} \cdot T_k. \tag{10}$$

where $d_k = \|\mathbf{p}_k - \mathbf{r}\|$ is the last-segment distance, $T_k = \prod_{m \in \mathcal{I}(k,\mathbf{r})} \tau_m$ is the occlusion product, and $\mathcal{I}(k, \mathbf{r})$ is the set of scatterers intersected by the segment from $\mathbf{p}_k$ to $\mathbf{r}$. The proof is in Appendix B.

This factorization has three corollaries that dictate the conditioning architecture, each following from the structure of η_{eff}, with proofs in Appendix C.

Corollary 1 (Global-local decomposition)**.** *Decomposing the propagation factor in Proposition 1 as $\eta_{\text{eff}}(\mathbf{p}_k, \mathbf{r}) = \bar{\eta}(\mathbf{r}) + \delta_k(\mathbf{r})$, where $\bar{\eta}(\mathbf{r}) = \frac{1}{K}\sum_k \eta_{\text{eff}}(\mathbf{p}_k, \mathbf{r})$ is the scatterer-averaged effect and $\delta_k(\mathbf{r})$ is the per-scatterer residual, Equation (9) becomes:*

$$\phi_k(\hat{\mathbf{d}}_k, \mathbf{r}) = \underbrace{\bar{\eta}(\mathbf{r}) \cdot \phi_k^{\text{base}}}_{\textit{global (shared)}} + \underbrace{\delta_k(\mathbf{r}) \cdot \phi_k^{\text{base}}}_{\textit{local (per-k)}}. \tag{11}$$

Since $\delta_k \neq 0$ for almost all $\mathbf{r}$, both branches are required.

Corollary 2 (Affine modulation suffices)**.** *The complex-valued product in Equation (11) can be represented by element-wise affine modulation $(1 + \boldsymbol{\alpha}) \odot \boldsymbol{\phi}_k^{\text{base}} + \boldsymbol{\beta}$ with $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{C}^d$, which jointly parameterizes the amplitude scaling and phase rotation induced by η_{eff}.*

Corollary 3 (Index-free form $\Rightarrow$ shared weights)**.** *η_{eff} depends on k only through the inputs (d_k, T_k), which permits a single shared network.*

4.3.2 Conditioning Architecture

We now instantiate the corollaries as architectural components. Given a receiver position $\mathbf{r}$, the conditioning function g_Θ proceeds in three steps: (i) $\mathbf{r}$ is mapped to a feature $\gamma(\mathbf{r})$ via a learnable Fourier encoding that captures the two-scale structure of η_{eff}. (ii) A global branch applies affine modulation to all Gaussians based on $\gamma(\mathbf{r})$, as required by Corollaries 1 and 2. (iii) A local branch refines each Gaussian individually using per-scatterer geometric and propagation features, as required by Corollary 1, with shared weights across all Gaussians by Corollary 3.

Receiver Encoding. The propagation factor η_{eff} in Equation (10) varies at two spatial scales: the path loss $\frac{1}{\|\mathbf{p}_k - \mathbf{r}\|}$ changes over room-scale distances, while the phase $e^{-j\frac{2\pi}{\lambda}\|\mathbf{p}_k - \mathbf{r}\|}$ oscillates at the centimeter-scale wavelength λ. We use a learnable Fourier encoding of $\mathbf{r}$ for both scales:

$$\gamma(\mathbf{r}) = \big[\sin(\omega_1^\top \mathbf{r}),\ \cos(\omega_1^\top \mathbf{r}),\ \ldots,\ \sin(\omega_F^\top \mathbf{r}),\ \cos(\omega_F^\top \mathbf{r})\big], \tag{12}$$

where the frequency vectors $\{\omega_f\}_{f=1}^{F}$ are initialized at logarithmically spaced scales to span both the room-scale path loss and the wavelength-scale phase, and are learned end-to-end so that they adapt to the scene-specific ratio between room geometry and operating wavelength.

Global Branch. The global branch captures shared receiver-dependence across all scatterers. When the receiver moves, the relative importance of directional harmonics shifts. Low-order components capture broad propagation patterns and are less sensitive to receiver position, while high-order components encode fine-grained multipath structure varying sharply with $\mathbf{r}$. To model this, an MLP generates per-component affine parameters from the receiver encoding and component identity:

$$(\boldsymbol{\alpha}_\ell,\, \boldsymbol{\beta}_\ell) = \mathrm{MLP}_{\mathrm{global}}([\gamma(\mathbf{r});\, \mathbf{c}_\ell;\, \mathbf{e}_\ell])\,, \tag{13}$$

where $\mathbf{c}_\ell$ encodes the structural identity of radiance component ℓ, *i.e.*, its degree and order in the Fourier–Legendre basis, and $\mathbf{e}_\ell$ is a learnable embedding. Since the radiance is complex-valued, $\boldsymbol{\alpha}_\ell$ and $\boldsymbol{\beta}_\ell$ operate on both real and imaginary parts, jointly modulating amplitude and phase as required by Corollary 2. The same modulation applies to all Gaussians:

$$\tilde{\phi}_k^{(\ell)}(\mathbf{r}) = (1 + \boldsymbol{\alpha}_\ell) \odot \phi_k^{\mathrm{base},(\ell)} + \boldsymbol{\beta}_\ell, \quad \forall\, k = 1, \ldots, K. \tag{14}$$

The total cost is $\mathcal{O}(L)$ independent of Gaussian count, where L is the number of radiance components, keeping the global branch lightweight even for scenes with many Gaussians.

Local Branch. The local branch captures per-scatterer variation that the global branch cannot represent. In RF propagation, a scatterer's contribution to the received signal depends critically on its individual relationship to the receiver. A wall in direct line-of-sight dominates the signal, while a scatterer behind multiple obstacles contributes little regardless of the global harmonic adjustment. To model this, we construct a six-dimensional feature for each Gaussian k that directly mirrors the three quantities governing η_{eff} in Equation (10): *direction*, *distance*, and *occlusion*.

The *direction* $\hat{\mathbf{v}}_k = \frac{\mathbf{r}-\mathbf{p}_k}{\|\mathbf{r}-\mathbf{p}_k\|}$ and *distance* $d_k = \|\mathbf{r} - \mathbf{p}_k\|$ are computed directly from the frozen Gaussian positions and the receiver position, capturing the free-space path loss $\frac{1}{d_k}$ and phase rotation $e^{-j\frac{2\pi}{\lambda}d_k}$ on the last segment. These features characterize the free-space component of η_{eff} per scatterer without additional learned geometry, since the Gaussian positions are frozen from Stage I.

The *occlusion*, however, requires the cumulative scene density along the path from $\mathbf{p}_k$ to $\mathbf{r}$, which we obtain from the frozen Gaussian geometry without additional supervision. At the start of Stage II, we voxelize the scene by splatting each Gaussian's transmittance into a 3D occupancy grid $\mathcal{V}$ based on its learned position $\mathbf{p}_k$ and covariance $\mathbf{C}_k$. For each Gaussian k, we cast a ray from $\mathbf{p}_k$ to $\mathbf{r}$ and probe the grid at S evenly spaced points $\{\mathbf{q}_{k,s}\}_{s=1}^{S}$:

$$T_k = \prod_{s=1}^{S} \big(1 - \mathcal{V}(\mathbf{q}_{k,s})\big), \qquad \bar{\rho}_k = \frac{1}{S}\sum_{s=1}^{S} \mathcal{V}(\mathbf{q}_{k,s}), \tag{15}$$

where $\mathcal{V}(\cdot)$ denotes the trilinearly interpolated occupancy. The transmittance T_k approximates the occlusion product $\prod_{m\in\mathcal{I}(k,\mathbf{r})} \tau_m$ from Equation (10): $T_k \to 1$ indicates a clear last segment, while $T_k \to 0$ indicates heavy occlusion. The mean density $\bar{\rho}_k$ provides complementary information, distinguishing a single concrete wall where T_k is low but $\bar{\rho}_k$ remains moderate, from distributed clutter where both T_k and $\bar{\rho}_k$ are high. A single shared MLP processes all Gaussians:

$$(\boldsymbol{\alpha}_k^{\mathrm{loc}},\, \boldsymbol{\beta}_k^{\mathrm{loc}}) = \mathrm{MLP}_{\mathrm{local}}([\hat{\mathbf{v}}_k;\, d_k;\, T_k;\, \bar{\rho}_k])\,. \tag{16}$$

Combined Modulation. The two branches compose sequentially: the global branch produces $\tilde{\phi}_k^{(\ell)}(\mathbf{r})$ via Equation (14), and the local branch then applies:

$$\phi_k(\hat{\mathbf{d}}_k, \mathbf{r}) = (1 + \boldsymbol{\alpha}_k^{\mathrm{loc}}) \odot \tilde{\phi}_k(\mathbf{r}) + \boldsymbol{\beta}_k^{\mathrm{loc}}, \tag{17}$$

mirroring the factored structure of η_{eff} in Equation (10). Both branches are zero-initialized. Consequently, the conditioning begins as an identity transformation and supports stable training from the Stage I solution. The total cost is $\mathcal{O}(L + K)$, and the occupancy grid is built once with no gradient overhead. At inference, g_Θ accepts receiver locations $\mathbf{r}^*$ excluded from training within the surveyed scene.

Table 1: Overall RF data synthesis performance. RxGS uses a single shared model for all N receivers. Note: GRaF reports spectrum results only, as its image-format backbone does not transfer to scalar RSSI or complex CSI without redesigning its entire architecture.

		BLE RSSI	**RFID Spectrum**			**WiFi CSI**
Method	#Models	MAE↓	MSE↓	PSNR↑	SSIM↑	SNR↑
NeRF^2 [5]	1	$3.40_{\pm 0.14}$	$0.0040_{\pm 0.0005}$	$24.58_{\pm 0.51}$	$0.868_{\pm 0.011}$	$21.00_{\pm 0.46}$
GRaF [33]	1	—	$0.0029_{\pm 0.0004}$	$26.40_{\pm 0.58}$	$0.903_{\pm 0.007}$	—
RF-3DGS [18]	N	$4.09_{\pm 0.66}$	$0.0054_{\pm 0.0004}$	$23.11_{\pm 0.32}$	$0.801_{\pm 0.008}$	$21.89_{\pm 0.56}$
Ours$_{\text{RF-3DGS}}$	1	$3.94_{\pm 0.48}$	$0.0068_{\pm 0.0007}$	$22.22_{\pm 0.38}$	$0.791_{\pm 0.012}$	$20.19_{\pm 0.39}$
GSRF [20]	N	$3.74_{\pm 0.41}$	$0.0052_{\pm 0.0004}$	$23.29_{\pm 0.35}$	$0.831_{\pm 0.010}$	$21.91_{\pm 0.52}$
Ours$_{\text{GSRF}}$	1	$3.57_{\pm 0.24}$	$0.0065_{\pm 0.0006}$	$22.47_{\pm 0.35}$	$0.780_{\pm 0.012}$	$21.12_{\pm 0.49}$
WRF-GS+ [19]	N	$5.84_{\pm 5.85}$	$0.0034_{\pm 0.0005}$	$25.81_{\pm 0.66}$	$0.892_{\pm 0.012}$	$21.84_{\pm 0.52}$
Ours$_{\text{WRF-GS+}}$	1	$6.17_{\pm 0.71}$	$0.0048_{\pm 0.0007}$	$24.27_{\pm 0.67}$	$0.860_{\pm 0.011}$	$20.55_{\pm 0.38}$

4.4 Single-Kernel Multi-Receiver Rasterization

Since the Stage I geometry is receiver-independent, per-Gaussian projection, depth sorting, and basis evaluation at $\hat{\mathbf{d}}_k$ depend only on the transmitter and are shared across all N receivers. For the FLE and SH backbones, receiver-specific radiance $\phi_k(\hat{\mathbf{d}}_k, \mathbf{r}_j)$ is evaluated in a batched operation after shared geometry work. Their rasterizer launches one CUDA kernel that tiles the rendering sphere along elevation θ and azimuth ϕ, with $\mathrm{gridDim} = (\mathrm{tiles}_\theta, \mathrm{tiles}_\phi, N)$. The receiver index maps to the third grid dimension, with each block reading its signal slice and writing its output slot. The shared projection and sorting work is amortized across receiver queries. The third grid dimension scales with N to evaluate receiver-specific radiance and write N outputs, as detailed in Appendix D.

5 Experiments

Implementation. We implement RxGS in PyTorch with custom CUDA kernels for RF signal tracing and rasterization. All experiments are conducted on a single NVIDIA H100 GPU. Full implementation and hyperparameter details are in Appendix E.

Datasets. We evaluate RxGS on three RF datasets: ❶ *BLE RSSI* [5], ❷ *RF Identification (RFID) spatial spectrum* (angular power distribution at the receiver) simulated via Sionna [45, 46], and ❸ *WiFi CSI* [47]. The datasets span scalar RSSI (21 receivers), spectrum matrices (21 receivers), and complex-valued CSI (8 receivers). Full dataset details are provided in Appendix F.

Metrics. We adopt metrics matched to each dataset's signal modality. ❶ *Mean absolute error (MAE)* in dBm for BLE RSSI. ❷ *Mean squared error (MSE)*, *peak signal-to-noise ratio (PSNR)*, and *structural similarity index (SSIM)* [48] for RFID spatial spectrum. ❸ *Signal-to-noise ratio (SNR)* in dB for WiFi CSI, computed across $N_{\text{sc}} = 26$ subcarriers:

$$\text{SNR} = -10 \log_{10}\left(\textstyle\sum_k |\hat{h}_k - h_k|^2 \big/ \sum_k |h_k|^2\right) \tag{18}$$

where h_k and $\hat{h}_k$ are the ground-truth and predicted complex CSI at subcarrier k. For each metric, we first compute the per-receiver mean by averaging over all test samples. We then report the mean and standard deviation across these per-receiver means. This convention emphasizes consistency across receivers rather than per-sample prediction variance.

Baselines. We compare RxGS against five baselines. ❶ *NeRF^2* [5] is an MLP-based neural RF radiance field, representing the leading NeRF-based approach. ❷ *GRaF* [33] is a NeRF-based cross-scene spectrum synthesis method that interpolates reference spectra from neighboring transmitters at the query receiver. ❸ *GSRF* [20] is a complex-valued FLE-based 3DGS method for RF synthesis, representing the non-neural 3DGS approach, trained per receiver. ❹ *RF-3DGS* [18] is the SH-based 3DGS approach for RF synthesis, also trained per receiver. Since camera-based geometry initialization is unavailable in RF-only settings, we adopt the geometry-learning pipeline of GSRF [20] and replace FLE with SH to match the RF-3DGS radiance formulation. ❺ *WRF-GS+* [19] adapts deformable 3DGS from the optical domain to RF synthesis by conditioning the deformation network on transmitter position, representing the leading neural 3DGS approach, also trained per receiver.

5.1 Overall Performance Against Per-Receiver Baselines

We evaluate RxGS against per-receiver baselines on two axes, synthesis quality in Table 1 and computational cost in Table 2. Appendices G.1–G.11 give receiver-level results and analyses.

Synthesis Quality. Table 1 shows the cost–accuracy tradeoff of one shared receiver-conditioned model: mean BLE RSSI error falls for two backbones and rises by 0.33 dBm for WRF-GS+, while spectrum and CSI trail per-receiver baselines by 0.8 to 1.7 dB. NeRF-based approaches achieve the highest spectrum quality, with GRaF [33] reaching 26.40 dB PSNR and NeRF^2 reaching 24.58 dB, but at prohibitive cost [18–20]. RxGS instead retains an explicit 3DGS representation to reduce training and inference costs.

On BLE RSSI, RxGS uses one shared model across receivers for each backbone. **Ours**$_{\text{GSRF}}$ improves MAE from 3.74 to 3.57 dBm, and **Ours**$_{\text{RF-3DGS}}$ from 4.09 to 3.94 dBm. The effect is most pronounced for WRF-GS+, whose per-receiver baseline has two high-error receivers above 18 dBm. **Ours**$_{\text{WRF-GS+}}$ removes those outliers, while mean MAE increases by 0.33 dBm. Appendix G.1 gives the per-receiver breakdown.

On RFID spectrum and WiFi CSI, RxGS trails per-receiver baselines by 0.8 to 1.7 dB. On spatial spectrum, the 90×360 angular signal has fine directional structure, and the shared variants trail their corresponding per-receiver 3DGS baselines by 0.8 to 1.5 dB PSNR. On CSI, the complex-valued FLE backbone **Ours**$_{\text{GSRF}}$ has the smallest gap among the evaluated 3DGS variants, dropping 0.79 dB from 21.91 to 21.12. Appendix G.2 provides qualitative CSI predictions tracking ground truth in amplitude and phase. The shared model also supports queries at fixed receivers excluded from training within the surveyed scene. These results hold across three architecturally distinct 3DGS backbones, *i.e.*, RF-3DGS [18], GSRF [20], and WRF-GS+ [19]. The three backbones use real SH, complex-valued FLE, and deformation networks, respectively. Our conditioning modules are compatible with each in the evaluated datasets.

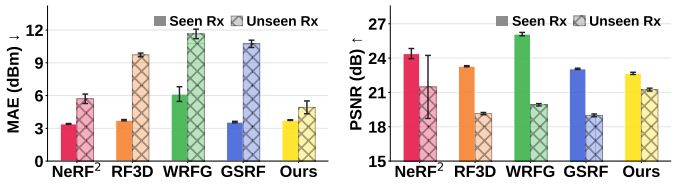
Table 2: Training, inference, and storage for serving $N = 21$ receivers on RFID.

Method	Train (h)	Infer. (ms)	Stor. (MB)
NeRF^2 [5]	14.3	2436.4	2.6
GRaF [33]	16.2	59891.4	143.7
RF-3DGS [18]	2.8	156.4	181.9
Ours$_{\text{RF-3DGS}}$	0.4	35.3	8.8
GSRF [20]	5.5	238.1	548.0
Ours$_{\text{GSRF}}$	0.6	31.2	26.1
WRF-GS+ [19]	13.6	52.4	75.6
Ours$_{\text{WRF-GS+}}$	0.3	42.5	3.2

Computational Cost. Table 2 summarizes end-to-end RFID costs at $N = 21$, with per-backbone details in Appendix G.3. RxGS accelerates training by $7\times$ to $45\times$ over per-receiver baselines by reducing N independent training runs to a single shared-model fit. Our multi-receiver rasterizer renders all N receivers in one batched CUDA call. At $N = 21$, per-transmitter inference is $7.6\times$ faster than per-receiver GSRF, $4.4\times$ faster than RF-3DGS, and $1.2\times$ faster than WRF-GS+. Compared with NeRF^2 and GRaF, the shared 3DGS variants are $60\times$ to $1900\times$ faster. Replacing N per-receiver checkpoints with one shared model compresses the deployable footprint by roughly $21\times$. The FLE and SH variants share projection and sorting across receiver queries. WRF-GS+ retains a per-receiver render loop and has a smaller inference gain.

5.2 Generalization to Unseen Receivers

We partition the 21 receivers into 3 folds of 7, train on 14, and evaluate on all 21 receivers. We exclude WiFi CSI because its 8 receivers are too few for meaningful seen/unseen splits. Per-receiver baselines use the nearest trained receiver's checkpoint at unseen positions, while receiver-conditioned methods are queried directly. GRaF [33] is excluded because it requires reference spectra from the query receiver.



(a) BLE RSSI (b) RFID Spectrum

Figure 2: Unseen-receiver generalization. "Ours" is **Ours**$_{\text{GSRF}}$, and RF3D / WRFG denote RF-3DGS [18] / WRF-GS+ [19].

Figure 2 reports the mean and standard deviation across folds. "Ours" denotes **Ours**$_{\text{GSRF}}$, the variant closest to its per-receiver baseline in Table 1.

For BLE RSSI, per-receiver GSRF, RF-3DGS, and WRF-GS+ rise from 3.6–6.1 dBm seen MAE to 9.7–11.6 dBm at unseen receivers. On RFID spectrum, they lose 4–6 dB PSNR. WRF-GS+ falls from 26.1 to 19.9 dB. Appendix G.4 shows examples.

Receiver Generalization. NeRF^2, which conditions a continuous neural field on the receiver position $\mathbf{r}$, degrades much more gracefully, moving from 3.40 to 5.70 dBm on BLE. RxGS achieves the lowest unseen MAE of 4.92 dBm on BLE while matching the best seen performance at 3.75 dBm. On spectrum, RxGS nearly matches NeRF^2 on unseen PSNR at 21.25 versus 21.48 dB, while staying within 2 dB of its own seen performance on both datasets. Appendix G.4 visualizes the predicted spectra: RxGS recovers sharp directional peaks at unseen receivers, NeRF^2 predictions are overly smooth, and WRF-GS+ collapses to the closest seen-receiver pattern. Receiver conditioning supports prediction at unseen fixed receiver positions within the surveyed scene.

5.3 Ablation Study

We ablate each architectural design on BLE RSSI with **Ours**$_{\text{GSRF}}$, training and evaluating on all 21 receivers. ❶ Joint training raises MAE from 3.57 to 4.44 dBm, the largest gap in the table, consistent with the decomposition derived in Appendix A: jointly optimizing geometry and radiance prevents geometry from converging to a receiver-independent solution. ❷ Removing the local or global branch raises MAE to 4.11 and 3.91 dBm, respectively, showing neither alone suffices as predicted by Corollary 1: the global branch captures the scatterer-averaged effect but cannot represent per-scatterer residuals, while the local branch captures residuals but cannot efficiently model the shared component. ❸ Replacing complex affine modulation $(1+\boldsymbol{\alpha}) \odot \boldsymbol{\phi}^{\text{base}} + \boldsymbol{\beta}$ with an additive shift $\boldsymbol{\phi}^{\text{base}} + \boldsymbol{\beta}$ raises MAE to 4.02 dBm, consistent with the analysis in Corollary 2: additive conditioning lacks the multiplicative structure needed to represent η_{eff}. ❹ Removing the occlusion features T_k and $\bar{\rho}_k$ raises MAE to 3.95 dBm, showing that the last-segment transmittance product contributes meaningful amplitude information beyond per-scatterer direction and distance.

Table 3: Ablation study on BLE RSSI.

Variant	MAE↓ (dBm)
Full RxGS – Ours$_{\text{GSRF}}$	$\mathbf{3.57}_{\pm 0.24}$
Joint training (no two-stage)	$4.44_{\pm 0.39}$
Global branch only	$4.11_{\pm 0.40}$
Local branch only	$3.91_{\pm 0.28}$
Additive modulation	$4.02_{\pm 0.36}$
Without occlusion features	$3.95_{\pm 0.25}$

The global-only branch provides an environment-map-like RF variant. Combining it with local conditioning reduces BLE MAE from 4.11 to 3.57 dBm. Appendix G.5 shows that accuracy at unseen receivers improves with more training receivers, with diminishing returns beyond about 12 in the evaluated scene. Appendix G.6 gives a 0.55 dBm spread across single-receiver Stage I reference choices, and Appendices G.10–G.11 report grid-resolution and conditioning-width sensitivity.

5.4 Application Study: Sparse-Deployment BLE Localization

Fingerprint-based indoor localization trades off accuracy against gateway deployment cost. Dense deployments provide high accuracy at high cost, whereas sparse deployments reduce cost at the expense of accuracy. We show that RxGS bridges this gap by synthesizing fingerprints at virtual gateway positions. Following the protocol of §5.2, we assume only $N=5$ gateways are physically installed and use RxGS to synthesize RSSI at the remaining 16 positions, forming a 21-dimensional fingerprint. The 16 virtual positions coincide with measured gateway sites excluded from training, allowing direct ground-truth validation of their synthesized fingerprints. Each of the 1,200 test transmitters is localized via weighted K-nearest-neighbor ($K=5$) against a database of 4,800 training positions.

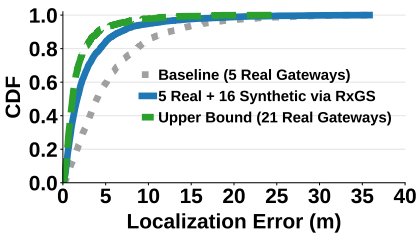


Figure 3: Localization: RxGS bridges 71% of the sparse-to-dense gap.

Figure 3 shows the cumulative distribution function (CDF) of localization error. The sparse 5-gateway baseline yields a mean error of 5.61 ± 0.23 m. Augmenting it with 16 RxGS-synthesized fingerprints reduces the mean error to 2.98 ± 0.67 m, a 47% improvement over the sparse baseline. This closes 71% of the gap to the full 21-gateway oracle, which achieves 1.91 m.

For 5, 8, 11, 14, and 17 physical gateways, mean errors over 20 random subsets are 5.51, 4.11, 3.13, 2.69, and 2.27 m, respectively. Table 4 gives 1.91 m for the full 21-gateway reference. The fixed five-gateway layout in Figure 3 yields 5.61 m, compared with 5.51 m averaged across the random layouts in Table 4. Appendix G.7 studies network planning.

Table 4: BLE localization error versus number of physical gateways. For 5–17 gateways, errors are averaged over 20 random subsets. The 21-gateway result uses all measurements.

Physical gateways	5	8	11	14	17	21
Mean error (m)↓	5.51	4.11	3.13	2.69	2.27	1.91

Table 4 shows smaller error reductions as physical gateways are added, motivating spatially diverse placement and synthesized fingerprints within the surveyed region.

6 Conclusion and Future Work

RxGS learns shared 3DGS geometry and receiver-conditioned radiance for fixed receivers in a surveyed scene. It predicts at receiver positions excluded from training using one model instead of N receiver-specific checkpoints. On RSSI, it improves RF-3DGS and GSRF, while its WRF-GS+ variant eliminates two high-error outliers with a 0.33 dBm increase in mean MAE. On spatial spectrum and CSI, it trails per-receiver baselines by 0.8 to 1.7 dB. Training, inference, and storage costs decrease by up to $45\times$, $7.6\times$, and $24\times$, respectively.

The method assumes a static surveyed scene, requires a reference-receiver survey in each new environment, and models last-segment receiver effects within the 3DGS rendering abstraction. Angular-spectrum experiments use simulated data, while measured CSI spans eight effective receivers in one room. Future work will evaluate measured multi-receiver spectra, larger and dynamic scenes, full-multipath variation, computational scaling, and cross-environment transfer.

Acknowledgments and Disclosures

The research reported in this paper was sponsored in part by the National Science Foundation (award # ECCS-2525614) and the U.S. Army DEVCOM Army Research Laboratory (award # W911NF1720196). Mani Srivastava was partially supported by the Mukund Padmanabhan Term Chair at UCLA. Kang Yang was partially supported by the UCLA Institute for Digital Research and Education (IDRE) fellowship. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the funding agencies. The authors declare no competing interests.

References

[1] Yongsen Ma, Gang Zhou, and Shuangquan Wang. WiFi Sensing with Channel State Information: A Survey. *ACM Computing Surveys*, 52(3):1–36, 2019.

[2] Ali Abedi, Farzan Dehbashi, Mohammad Hossein Mazaheri, Omid Abari, and Tim Brecht. WiTAG: Seamless WiFi Backscater Communication. In *ACM Special Interest Group on Data Communications (SIGCOMM)*, 2020.

[3] Theodore S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, 2002.

[4] Ryu Okubo, Luke Jacobs, Jinhua Wang, Steven Bowers, and Elahe Soltanaghai. Integrated Two-Way Radar Backscatter Communication and Sensing with Low-Power IoT Tags. In *ACM Special Interest Group on Data Communications (SIGCOMM)*, 2024.

[5] Xiaopeng Zhao, Zhenlin An, Qingrui Pan, and Lei Yang. NeRF^2: Neural Radio-Frequency Radiance Fields. In *ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2023.

[6] Jiachen Lu, Hailan Shanbhag, and Haitham Al Hassanieh. GeRaF: Neural Geometry Reconstruction from Radio Frequency Signals. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

[7] Chaitanya Amballa, Sattwik Basu, Yu-Lin Wei, Zhijian Yang, Mehmet Ergezer, and Romit Roy Choudhury. Can NeRFs See without Cameras? *arXiv preprint arXiv:2505.22441*, 2025.

[8] Liuyi Yang, Patrick Finnerty, and Chikara Ohta. Diffusion Model-based RSSI Fingerprint Generation for Indoor Localization in Dynamic Environments. In *IEEE Wireless and Optical Communications Conference (WOCC)*, pages 402–407, 2025.

[9] Kang Yang, Yuning Chen, and Wan Du. OrchLoc: In-Orchard Localization via a Single LoRa Gateway and Generative Diffusion Model-Based Fingerprinting. In *ACM International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2024.

[10] Pushpendu Kar and Bhasker Dappuri. Site Survey and Radio Frequency Planning for the Deployment of Next Generation WLAN. In *Topical Conference on Wireless Sensors and Sensor Networks (WiSNet)*, 2018.

[11] Cisco. Understand Site Survey Guidelines for WLAN Deployment. `https://www.cisco.com/c/en/us/support/docs/wireless/5500-series-wireless-controllers/116057-site-survey-guidelines-wlan-00.html`, 2023. [Online].

[12] Hyojeong Shin, Yohan Chon, Yungeun Kim, and Hojung Cha. MRI: Model-Based Radio Interpolation for Indoor War-Walking. *IEEE Transactions on Mobile Computing*, 14(6):1231–1244, 2014.

[13] Felipe Parralejo, Fernando J. Aranda, José A. Paredes, Fernando J. Álvarez, and Jorge Morera. Comparative Study of Different BLE Fingerprint Reconstruction Techniques. In *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2021.

[14] Zhengqing Yun and Magdy F Iskander. Ray Tracing for Radio Propagation Modeling: Principles and Applications. *IEEE Access*, 3:1089–1100, 2015.

[15] Esteban Egea-Lopez, Jose Maria Molina-Garcia-Pardo, Martine Lienard, and Pierre Degauque. Opal: An Open Source Ray-Tracing Propagation Simulator for Electromagnetic Characterization. *Plos One*, 16(11):e0260060, 2021.

[16] MATLAB. Indoor MIMO-OFDM Communication Link Using Ray Tracing. `https://www.mathworks.com/help/comm/ug/indoor-mimo-ofdm-communication-link-using-ray-tracing.html`, 2023. [Online].

[17] Lidong Guo, Xuefei Ning, Yonggan Fu, Tianchen Zhao, Zhuoliang Kang, Jincheng Yu, Yingyan Celine Lin, and Yu Wang. Rad-NeRF: Ray-Decoupled Training of Neural Radiance Field. *Conference on Neural Information Processing Systems (NeurIPS)*, 37:113742–113771, 2024.

[18] Lihao Zhang, Haijian Sun, Samuel Berweger, Camillo Gentile, and Rose Qingyang Hu. RF-3DGS: Wireless Channel Modeling with Radio Radiance Field and 3D Gaussian Splatting. *IEEE Journal on Selected Areas in Communications*, 2024.

[19] Chaozheng Wen, Jingwen Tong, Yingdong Hu, Zehong Lin, and Jun Zhang. WRF-GS+: Wireless Radiation Field Reconstruction with 3D Gaussian Splatting. *IEEE Transactions on Wireless Communications*, 2025.

[20] Kang Yang, Gaofeng Dong, Sijie Ji, Wan Du, and Mani Srivastava. GSRF: Complex-Valued 3D Gaussian Splatting for Efficient Radio-Frequency Data Synthesis. *arXiv preprint arXiv:2502.01826*, 2025.

[21] Haofan Lu, Christopher Vattheuer, Baharan Mirzasoleiman, and Omid Abari. NeWRF: A Deep Learning Framework for Wireless Radiation Field Reconstruction and Channel Prediction. In *International Conference on Machine Learning (ICML)*, 2024.

[22] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. In *ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, 2023.

[23] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-Splatting: Alias-Free 3D Gaussian Splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[24] Guikun Chen and Wenguan Wang. A Survey on 3D Gaussian Splatting. *arXiv preprint arXiv:2401.03890*, 2024.

[25] Robin Chataut and Robert Akl. Massive MIMO Systems for 5G and beyond Networks—overview, Recent Trends, Challenges, and Future Research Direction. *Sensors*, 20(10):2753, 2020.

[26] Mahnoor Ajmal, Ayesha Siddiqa, Bomi Jeong, Junho Seo, and Dongkyun Kim. Cell-Free Massive Multiple-Input Multiple-Output Challenges and Opportunities: A Survey. *ICT Express*, 10(1):194–212, 2024.

[27] Volker Schönefeld. Spherical Harmonics. *Computer Graphics and Multimedia Group*, 18, 2005.

[28] Rasmus Cornelius and Dirk Heberling. Spherical Wave Expansion with Arbitrary Origin for Near-Field Antenna Measurements. *IEEE Transactions on Antennas and Propagation*, 65(8):4385–4388, 2017.

[29] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[30] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. MVS-NeRF: Fast Generalizable Radiance Field Reconstruction from Multi-View Stereo. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[31] Jia-Wei Liu, Yan-Pei Cao, Tianyuan Yang, Zhongcong Xu, Jussi Keppo, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Hosnerf: Dynamic human-object-scene neural radiance fields from a single video. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 18437–18448. IEEE, 2023.

[32] Tiantian Wang, Xinxin Zuo, Fangzhou Mu, Jian Wang, and Ming-Hsuan Yang. Towards 4D human video stylization. *Computer Vision and Image Understanding (CVIU)*, 262:104532, 2025.

[33] Kang Yang, Yuning Chen, and Wan Du. Generalizable Radio-Frequency Radiance Fields for Spatial Spectrum Synthesis. *arXiv preprint arXiv:2502.05708*, 2025.

[34] Kang Yang, Duaa Nakshbandi, Wan Du, and Mani Srivastava. CoRF: Cross-Scene RF Synthesis by Learning Propagation and Preserving Array Physics. *arXiv preprint arXiv:2609.34399*, 2026.

[35] Jingzhou Shen, Tianya Zhao, Yanzhao Wu, and Xuyu Wang. NeRF-APT: A New NeRF Framework for Wireless Channel Prediction. In *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2025.

[36] Zihang Zeng, Shu Sun, Meixia Tao, Yin Xu, and Xianghao Yu. VoxelRF: Voxelized Radiance Field for Fast Wireless Channel Modeling. *IEEE Communications Letters*, 30:617–621, 2025.

[37] Mufan Liu, Cixiao Zhang, Qi Yang, Yujie Cao, Yiling Xu, Yin Xu, Shu Sun, Mingzeng Dai, and Yunfeng Guan. Deformable 2D Gaussian Splatting for Efficient Wireless Radiance Field Rendering. *IEEE Transactions on Visualization and Computer Graphics*, pages 1–17, 2026.

[38] Kang Yang, Wan Du, and Mani Srivastava. Scalable 3D Gaussian Splatting-Based RF Signal Spatial Propagation Modeling. In *ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2025.

[39] Ron Levie, Cagkan Yapar, Gitta Kutyniok, and Giuseppe Caire. RadioUNet: Fast Radio Map Estimation With Convolutional Neural Networks. *IEEE Transactions on Wireless Communications*, 2021.

[40] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. NeRD: Neural Reflectance Decomposition from Image Collections. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[41] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. NeRFactor: Neural Factorization of Shape and Reflectance under an Unknown Illumination. *ACM Transactions on Graphics*, 40(6):1–18, 2021.

[42] Jian Gao, Chun Gu, Youtian Lin, Zhihao Li, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3D Gaussians: Realistic Point Cloud Relighting with Brdf Decomposition and Ray Tracing. In *European Conference on Computer Vision (ECCV)*, 2024.

[43] Zhihao Liang, Qi Zhang, Ying Feng, Ying Shan, and Kui Jia. GS-IR: 3D Gaussian Splatting for Inverse Rendering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[44] Shunsuke Saito, Gabriel Schwartz, Tomas Simon, Junxuan Li, and Giljoo Nam. Relightable Gaussian Codec Avatars. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[45] Jakob Hoydis, Sebastian Cammerer, Fayçal Ait Aoudia, Avinash Vem, Nikolaus Binder, Guillermo Marcus, and Alexander Keller. Sionna: An Open-Source Library for Next-Generation Physical Layer Research. *arXiv preprint arXiv:2203.11854*, 2022.

[46] Jakob Hoydis, Fayçal Aït Aoudia, Sebastian Cammerer, Merlin Nimier-David, Nikolaus Binder, Guillermo Marcus, and Alexander Keller. Sionna RT: Differentiable Ray Tracing for Radio Propagation Modeling. In *IEEE Globecom Workshops (GC Wkshps)*, 2023.

[47] Sibren De Bast and Sofie Pollin. Ultra Dense Indoor MaMIMO CSI Dataset. `https://dx.doi.org/10.21227/nr6k-8r78`, 2025. [Online].

[48] A Hore and D Ziou. Image Quality Metrics: PSNR Vs. SSIM. International Conference on Pattern Recognition. In *International Conference on Pattern Recognition (ICPR)*, 2010.

[49] Clayton Shepard, Jian Ding, Ryan E Guerra, and Lin Zhong. Understanding Real Many-Antenna MU-MIMO Channels. In *Asilomar Conference on Signals, Systems and Computers (ACSCC)*, 2016.

Appendix Contents

A Proof of Geometry-Radiance Decomposition **14**

B Proof of Last-Segment Factorization **16**

C Proofs of Corollaries 1–3 **17**

C.1 Proof of Corollary 1 17

C.2 Proof of Corollary 2 17

C.3 Proof of Corollary 3 18

D Single-Kernel Multi-Receiver Rasterization **18**

E Implementation Details **19**

E.1 Scene Representation 20

E.2 Signal Rendering Pipeline 20

E.3 Loss Function 21

E.4 Stage I: Geometry Learning 21

E.5 Stage II: Receiver-Conditioned Radiance 21

F Dataset Details **22**

F.1 BLE RSSI 22

F.2 RFID Spatial Spectrum 22

F.3 WiFi CSI 23

G Additional Experiments **24**

G.1 Per-Receiver Performance Breakdown 24

G.2 Qualitative CSI Trace Visualization 26

G.3 Computational Cost: Detailed Breakdown 26

G.4 Qualitative Spectrum Visualization 28

G.5 Scaling with Number of Receivers 29

G.6 Sensitivity to Reference Receiver Choice 30

G.7 Application Study: Network Planning 30

G.8 Receiver-Level Accuracy Gaps 31

G.9 Visibility, Bounce Depth, and Occlusion Features 31

G.10 Occupancy-Grid Resolution 32

G.11 Conditioning Capacity 32

H Four Evaluation Configurations **32**

I Broader Impacts **33**

A Proof of Geometry-Radiance Decomposition

We prove that in the 3DGS-based RF rendering framework, the receiver position $\mathbf{r}$ influences the rendered signal only through the radiance coefficients, while all remaining attributes are receiver-independent.

Notation. We use $\mathbf{s}_k$ for scatterer positions in this appendix to distinguish the physical scatterer from the Gaussian mean $\mathbf{p}_k$. Under Assumption 3, $\mathbf{s}_k$ and $\mathbf{p}_k$ coincide, so the two symbols are interchangeable.

Definition 1 (Multipath Channel Model)**.** *The received signal at receiver* $\mathbf{r}$ *from transmitter* $\mathbf{t}$ *in a scene with* K *scatterers at fixed positions* $\{\mathbf{s}_k\}_{k=1}^K$ *is:*

$$S(\mathbf{t},\mathbf{r}) = \sum_{l=1}^{L} h_l(\mathbf{t},\mathbf{r}), \tag{19}$$

where L *is the number of propagation paths, each traversing an ordered sequence of* $P_l \geq 0$ *scatterers* $(\mathbf{s}_{l,1},\ldots,\mathbf{s}_{l,P_l})$*. A direct line-of-sight path corresponds to* $P_l = 0$ *with a single segment* $d_{l,0} = \|\mathbf{t}-\mathbf{r}\|$*. The complex coefficient of the* l*-th path is:*

$$h_l(\mathbf{t},\mathbf{r}) = \underbrace{\prod_{i=1}^{P_l} \Gamma(\mathbf{s}_{l,i})}_{\text{reflection coefficients}} \cdot \underbrace{\prod_{j=0}^{P_l} \frac{\lambda}{4\pi d_{l,j}}}_{\text{free-space path loss}} \cdot \underbrace{e^{-j\frac{2\pi}{\lambda}\sum_{j=0}^{P_l} d_{l,j}}}_{\text{phase accumulation}}, \tag{20}$$

with segment distances $d_{l,0} = \|\mathbf{t}-\mathbf{s}_{l,1}\|$*,* $d_{l,j}$ *between consecutive scatterers, and* $d_{l,P_l} = \|\mathbf{s}_{l,P_l}-\mathbf{r}\|$ *for* $P_l \geq 1$*.*

Definition 2 (RF Rendering Equation)**.** *Given 3D Gaussians* $\{\mathcal{G}_k\}_{k=1}^K$ *with attributes* $\mathcal{G}_k = (\mathbf{p}_k, \mathbf{C}_k, \phi_k, \tau_k)$*, the rendered signal for a ray in direction* $\boldsymbol{\omega}$ *emitted from* $\mathbf{t}$ *is:*

$$R(\boldsymbol{\omega}) = \sum_{k=1}^{K_{\boldsymbol{\omega}}} g_k(\boldsymbol{\omega};\, \mathbf{p}_k, \mathbf{C}_k) \cdot \phi_k\left(\hat{\mathbf{d}}_k\right) \cdot \prod_{m=1}^{k-1} \tau_m, \tag{21}$$

where g_k *is the Gaussian weight measuring ray-Gaussian overlap,* $\phi_k\left(\hat{\mathbf{d}}_k\right) \in \mathbb{C}$ *is the directional radiance along* $\hat{\mathbf{d}}_k = \frac{\mathbf{p}_k-\mathbf{t}}{\|\mathbf{p}_k-\mathbf{t}\|}$*, and* $\prod_{m=1}^{k-1}\tau_m$ *is the cumulative transmittance along the ray. The transmittance* τ_k *models intrinsic material attenuation at* $\mathbf{s}_k$*; directional path-length effects are carried by* g_k*. The total rendered signal* $\hat{S}$ *aggregates* $R(\boldsymbol{\omega})$ *over a finite set of sampled ray directions, and represents all signal contributions at the receiver, including the direct line-of-sight component absorbed by the Gaussians along the* $\mathbf{t}\to\mathbf{r}$ *line.*

Definition 3 (Receiver-Independence)**.** *A scene attribute* ξ *is* receiver-independent *if* $\frac{\partial \xi}{\partial \mathbf{r}} = \mathbf{0}$ *for all* $\mathbf{r}$*, and* receiver-dependent *if* $\xi(\mathbf{r}_1) \neq \xi(\mathbf{r}_2)$ *for some* $\mathbf{r}_1, \mathbf{r}_2$*.*

Assumption 1 (Static scene)**.** *Scatterer positions* $\{\mathbf{s}_k\}$*, their spatial extents, and material properties* $\{\Gamma(\mathbf{s}_k)\}$ *are fixed and do not depend on* $\mathbf{t}$ *or* $\mathbf{r}$*.*

Assumption 2 (Non-triviality)**.** *There exist* $\mathbf{r}_1 \neq \mathbf{r}_2$ *and* $\mathbf{t}$ *such that* $S(\mathbf{t},\mathbf{r}_1) \neq S(\mathbf{t},\mathbf{r}_2)$*.*

Assumption 3 (Per-scatterer modeling abstraction)**.** *Each Gaussian* $\mathcal{G}_k$ *is treated as representing the aggregate signal contribution localized at a scene point* $\mathbf{s}_k$*, with the Gaussian weight* W_k *taken to absorb the cumulative propagation and attenuation effects of all paths whose last interaction is at* $\mathbf{s}_k$*, together with the direct line-of-sight contribution for Gaussians along the* $\mathbf{t}\to\mathbf{r}$ *line. This is a modeling abstraction adopted by 3DGS-based RF methods to bridge the straight-ray rendering pipeline and the bouncing-path multipath model; under this abstraction,* W_k *is identified with the multipath aggregate* Φ_k *in Equation* (22)*, as discussed in Remark 1.*

Assumption 4 (Rendering-model representability)**.** *At the learned optimum,* $\hat{S}(\mathbf{t},\mathbf{r}) = S(\mathbf{t},\mathbf{r})$ *in the limit of perfect representability.*

Proposition 2 (Geometry-Radiance Decomposition)**.** *Under Assumptions 1–4:*

1. *The geometric attributes* $\{\mathbf{p}_k, \mathbf{C}_k, \tau_k\}$ *are receiver-independent.*

2. *The radiance* $\phi_k = \phi_k\left(\hat{\mathbf{d}}_k, \mathbf{r}\right)$ *is receiver-dependent, with receiver-gradient given by Equation* (27) *under the per-scatterer identification.*

3. *Under the per-scatterer identification, the radiance is the only term in Equation* (21) *that carries receiver-dependence; all remaining factors are receiver-independent.*

Proof. **Step 1 (Per-scatterer decomposition).** Each propagation path terminates at the receiver, with its final segment originating from a scatterer $\mathbf{s}_{l,P_l}$ for $P_l \geq 1$, or directly from $\mathbf{t}$ for the line-of-sight path $P_l = 0$. Let $\mathcal{L}_k$ denote the set of paths whose last interaction is localized at $\mathbf{s}_k$, including the LoS path assigned to the Gaussian nearest the $\mathbf{t} \to \mathbf{r}$ line under Assumption 3. The sets $\{\mathcal{L}_k\}$ partition all paths, yielding:

$$S(\mathbf{t}, \mathbf{r}) = \sum_{k=1}^{K} \Phi_k(\mathbf{t}, \mathbf{r}), \qquad \Phi_k \triangleq \sum_{l \in \mathcal{L}_k} h_l(\mathbf{t}, \mathbf{r}). \tag{22}$$

For the rendering equation, define the per-ray weight $w_k(\boldsymbol{\omega}) = g_k(\boldsymbol{\omega}; \mathbf{p}_k, \mathbf{C}_k) \prod_{m=1}^{k-1} \tau_m$. Aggregating over rays:

$$\hat{S}(\mathbf{t}, \mathbf{r}) = \sum_{k=1}^{K} W_k \cdot \phi_k\left(\hat{\mathbf{d}}_k, \mathbf{r}\right), \qquad W_k \triangleq \sum_{\boldsymbol{\omega}} w_k(\boldsymbol{\omega}). \tag{23}$$

Under Assumptions 3 and 4, the per-scatterer identification is:

$$W_k \cdot \phi_k\left(\hat{\mathbf{d}}_k, \mathbf{r}\right) = \Phi_k(\mathbf{t}, \mathbf{r}), \quad \forall\, k. \tag{24}$$

This identification is the formal bridge between the rendering-equation weight W_k and the multipath aggregate Φ_k, and is the only point at which the two models are joined.

Step 2 (Geometric attributes are receiver-independent). By Assumption 1, $\mathbf{p}_k$, $\mathbf{C}_k$, and $\mathbf{s}_k$ are fixed, and $\boldsymbol{\omega}$ (defined on the emission surface centered at $\mathbf{t}$) is independent of $\mathbf{r}$. Chain rule gives $\partial g_k / \partial \mathbf{r} = \mathbf{0}$ and $\partial \tau_k / \partial \mathbf{r} = 0$, so:

$$\frac{\partial W_k}{\partial \mathbf{r}} = \mathbf{0}, \qquad \frac{\partial \hat{\mathbf{d}}_k}{\partial \mathbf{r}} = \mathbf{0}. \tag{25}$$

This establishes claim 1.

Step 3 (Physical form of the radiance receiver-gradient). Under the per-scatterer identification, Equation (24) with $W_k \neq 0$ gives $\phi_k = \Phi_k / W_k$. Since W_k is receiver-independent:

$$\frac{\partial \phi_k}{\partial \mathbf{r}} = \frac{1}{W_k} \sum_{l \in \mathcal{L}_k} \frac{\partial h_l}{\partial \mathbf{r}}. \tag{26}$$

From Equation (20), h_l depends on $\mathbf{r}$ only through the last-segment distance $d_{l,P_l} = \|\mathbf{o}_l - \mathbf{r}\|$, where $\mathbf{o}_l = \mathbf{s}_{l,P_l}$ is the last scatterer for $P_l \geq 1$ and $\mathbf{o}_l = \mathbf{t}$ for the line-of-sight path $P_l = 0$. The chain rule with $\partial d_{l,P_l} / \partial \mathbf{r} = -(\mathbf{o}_l - \mathbf{r}) / \|\mathbf{o}_l - \mathbf{r}\|$ and $\partial h_l / \partial d_{l,P_l} = h_l\,(-1/d_{l,P_l} - j2\pi/\lambda)$ yields:

$$\frac{\partial \phi_k}{\partial \mathbf{r}} = \frac{1}{W_k} \sum_{l \in \mathcal{L}_k} h_l \cdot \left(\frac{1}{d_{l,P_l}} + \frac{j2\pi}{\lambda}\right) \cdot \frac{\mathbf{o}_l - \mathbf{r}}{\|\mathbf{o}_l - \mathbf{r}\|}. \tag{27}$$

The $1/d_{l,P_l}$ term is the logarithmic derivative of free-space path loss, and $2\pi/\lambda$ is the phase accumulation rate, both standard for last-leg propagation under the Friis law.

Step 4 (Radiance carries all receiver-dependence). By Assumptions 4 and 2, there exist $\mathbf{r}_1 \neq \mathbf{r}_2$ and $\mathbf{t}$ with $\hat{S}(\mathbf{t}, \mathbf{r}_1) \neq \hat{S}(\mathbf{t}, \mathbf{r}_2)$. By Equation (23) and the receiver-independence of W_k and $\hat{\mathbf{d}}_k$ from Step 2:

$$\sum_{k=1}^{K} W_k \cdot \left[\phi_k\left(\hat{\mathbf{d}}_k, \mathbf{r}_1\right) - \phi_k\left(\hat{\mathbf{d}}_k, \mathbf{r}_2\right)\right] \neq 0. \tag{28}$$

Hence there exists $k^\star$ with $W_{k^\star} \neq 0$ and $\phi_{k^\star}\left(\hat{\mathbf{d}}_{k^\star}, \mathbf{r}_1\right) \neq \phi_{k^\star}\left(\hat{\mathbf{d}}_{k^\star}, \mathbf{r}_2\right)$, establishing claim 2. Since Step 2 shows all other factors in Equation (21) are receiver-independent, the radiance is the only term that can carry receiver-dependence, establishing claim 3.

Combining the three claims, the rendering equation factorizes as:

$$\hat{S}(\mathbf{t},\mathbf{r}) = \mathcal{R}\left(\mathbf{t},\ \underbrace{\{\mathbf{p}_k, \mathbf{C}_k, \tau_k\}}_{\partial/\partial\,\mathbf{r}=\mathbf{0}},\ \underbrace{\left\{\phi_k\left(\hat{\mathbf{d}}_k, \mathbf{r}\right)\right\}}_{\partial/\partial\,\mathbf{r}\neq\mathbf{0}}\right), \tag{29}$$

which justifies freezing the geometric attributes after Stage I and conditioning only the radiance on $\mathbf{r}$ in Stage II. □

Remark 1 (Scope of Assumption 3)**.** *The straight-ray rendering equation and the bouncing-path multipath model are not structurally identical: single-bounce paths align naturally with Gaussians along $\mathbf{t} \to \mathbf{s}_k$, while multi-bounce and line-of-sight paths do not. Assumption 3 adopts the phenomenological abstraction by which the 3DGS-RF framework bridges these models, assigning each path to the Gaussian whose localized region captures its final signal contribution. This assumption is invoked only to derive the explicit gradient form in Step 3. Claims 1, 2 (existence), and 3 rest only on Assumptions 1, 2, and 4, and hold regardless of whether the per-scatterer abstraction is accepted. We frame the result as a structural decomposition of the rendering equation rather than a physical theorem. Its role in the paper is to motivate the two-stage architecture rather than to make claims about the underlying electromagnetic propagation.*

B Proof of Last-Segment Factorization

Proof. Equation (24) with $W_k \neq 0$ gives $\phi_k = \Phi_k / W_k$, and substituting $\Phi_k = \sum_{l\in\mathcal{L}_k} h_l$:

$$\phi_k(\hat{\mathbf{d}}_k, \mathbf{r}) = \frac{1}{W_k} \sum_{l\in\mathcal{L}_k} h_l(\mathbf{t}, \mathbf{r}). \tag{30}$$

By the definition of $\mathcal{L}_k$ (paths whose last interaction is at $\mathbf{s}_k$), the last scatterer is $\mathbf{s}_{l,P_l} = \mathbf{s}_k$ for all $l \in \mathcal{L}_k$, so the last-segment distance $d_{l,P_l} = \|\mathbf{s}_k - \mathbf{r}\|$ is shared across all paths in $\mathcal{L}_k$. Expanding h_l from Equation (20) and separating the last-segment terms:

$$h_l(\mathbf{t},\mathbf{r}) = \underbrace{\prod_{i=1}^{P_l} \Gamma(\mathbf{s}_{l,i}) \cdot \prod_{j=0}^{P_l-1} \frac{\lambda}{4\pi d_{l,j}} \cdot e^{-j\frac{2\pi}{\lambda}\sum_{j=0}^{P_l-1} d_{l,j}}}_{h_l^{\mathrm{up}}(\mathbf{t})} \cdot \underbrace{\frac{\lambda}{4\pi\|\mathbf{s}_k - \mathbf{r}\|} \cdot e^{-j\frac{2\pi}{\lambda}\|\mathbf{s}_k-\mathbf{r}\|}}_{\eta(\mathbf{s}_k,\mathbf{r})}, \tag{31}$$

where $h_l^{\mathrm{up}}(\mathbf{t})$ contains all reflection coefficients, path losses, and phase accumulations from the transmitter to scatterer k along path l. Since $\eta(\mathbf{s}_k, \mathbf{r})$ is identical for all $l \in \mathcal{L}_k$, it factors out of the sum:

$$\phi_k(\hat{\mathbf{d}}_k, \mathbf{r}) = \eta(\mathbf{s}_k, \mathbf{r}) \cdot \underbrace{\frac{1}{W_k} \sum_{l\in\mathcal{L}_k} h_l^{\mathrm{up}}(\mathbf{t})}_{\phi_k^{\mathrm{base}}(\hat{\mathbf{d}}_k,\mathbf{t})}. \tag{32}$$

This establishes the free-space factorization $\phi_k = \eta \cdot \phi_k^{\mathrm{base}}$.

When the last segment from $\mathbf{s}_k$ to $\mathbf{r}$ traverses intervening scatterers, the cumulative transmittance $\prod_{m\in\mathcal{I}(k,\mathbf{r})} \tau_m$ from the rendering equation (Equation (21)) attenuates the signal further. Under Assumption 3, this attenuation acts only on the last segment, so it multiplies η without affecting ϕ_k^{base}, yielding the effective propagation factor:

$$\eta_{\mathrm{eff}}(\mathbf{s}_k, \mathbf{r}) = \eta(\mathbf{s}_k, \mathbf{r}) \cdot \prod_{m\in\mathcal{I}(k,\mathbf{r})} \tau_m. \tag{33}$$

The full factorization follows:

$$\phi_k(\hat{\mathbf{d}}_k, \mathbf{r}) = \eta_{\mathrm{eff}}(\mathbf{s}_k, \mathbf{r}) \cdot \phi_k^{\mathrm{base}}(\hat{\mathbf{d}}_k, \mathbf{t}). \tag{34}$$

□

C Proofs of Corollaries 1–3

Notation follows Appendix A: $\mathbf{s}_k$ denotes scatterer positions, interchangeable with the Gaussian mean $\mathbf{p}_k$ under Assumption 3.

C.1 Proof of Corollary 1

Proof. Define the mean propagation effect and per-scatterer residual:

$$\bar{\eta}(\mathbf{r}) = \frac{1}{K}\sum_{k=1}^{K} \eta_{\text{eff}}(\mathbf{s}_k, \mathbf{r}), \qquad \delta_k(\mathbf{r}) = \eta_{\text{eff}}(\mathbf{s}_k, \mathbf{r}) - \bar{\eta}(\mathbf{r}). \tag{35}$$

By construction, $\eta_{\text{eff}} = \bar{\eta} + \delta_k$, and substituting into Proposition 1 yields the decomposition in Equation (11).

It remains to show $\delta_k \neq 0$ for almost all $\mathbf{r}$. By Equation (10), $\eta_{\text{eff}}(\mathbf{s}_{k_1}, \mathbf{r}) = \eta_{\text{eff}}(\mathbf{s}_{k_2}, \mathbf{r})$ requires (in the best case where occlusion terms match) $\|\mathbf{s}_{k_1} - \mathbf{r}\| = \|\mathbf{s}_{k_2} - \mathbf{r}\|$, whose solution set is the perpendicular bisector hyperplane of $\mathbf{s}_{k_1}$ and $\mathbf{s}_{k_2}$ — a set of Lebesgue measure zero in $\mathbb{R}^3$. Therefore, $\eta_{\text{eff}}(\mathbf{s}_{k_1}, \mathbf{r}) \neq \eta_{\text{eff}}(\mathbf{s}_{k_2}, \mathbf{r})$ for almost all $\mathbf{r}$, so $\delta_k \neq 0$ in general. The global term $\bar{\eta}(\mathbf{r}) \cdot \phi_k^{\text{base}}$ alone cannot capture per-scatterer variation, and the local term $\delta_k(\mathbf{r}) \cdot \phi_k^{\text{base}}$ alone cannot capture the shared component, so both branches are required. □

C.2 Proof of Corollary 2

Proof. From Proposition 1, $\phi_k\left(\hat{\mathbf{d}}_k, \mathbf{r}\right) = \eta_{\text{eff}}(\mathbf{s}_k, \mathbf{r}) \cdot \boldsymbol{\phi}_k^{\text{base}}\left(\hat{\mathbf{d}}_k, \mathbf{t}\right)$, where $\eta_{\text{eff}} \in \mathbb{C}$. We show that affine modulation suffices to represent this complex-valued multiplication, and that two simpler alternatives are insufficient in general.

Step 1: Complex multiplication as a structured linear map. Write $\eta_{\text{eff}} = a + jb$ and let $\phi_k^{\text{base},(n)} = x_n + jy_n$ denote the n-th component of the complex-valued radiance vector. Component-wise, the product $\eta_{\text{eff}} \cdot \phi_k^{\text{base},(n)}$ expands as:

$$\begin{bmatrix} \text{Re}(\phi_k^{(n)}) \\ \text{Im}(\phi_k^{(n)}) \end{bmatrix} = \underbrace{\begin{bmatrix} a & -b \\ b & a \end{bmatrix}}_{\mathbf{M}(\eta_{\text{eff}})} \begin{bmatrix} x_n \\ y_n \end{bmatrix}. \tag{36}$$

The matrix $\mathbf{M}$ simultaneously scales amplitude by $|\eta_{\text{eff}}| = \sqrt{a^2 + b^2}$ and rotates phase by $\angle\eta_{\text{eff}} = \text{atan2}(b, a)$. Any conditioning mechanism must be able to represent this joint amplitude-and-phase operation.

Step 2: Receiver-shared additive conditioning is insufficient. Consider an additive model $\phi_k = \phi_k^{\text{base}} + \boldsymbol{\beta}(\mathbf{r})$, where $\boldsymbol{\beta}$ depends on $\mathbf{r}$ only (shared across all Gaussians, with no per- k dependence). For the true radiance $\phi_k = \eta_{\text{eff}} \cdot \phi_k^{\text{base}}$, this requires:

$$\boldsymbol{\beta}(\mathbf{r}) = (\eta_{\text{eff}}(\mathbf{s}_k, \mathbf{r}) - 1) \cdot \phi_k^{\text{base}}. \tag{37}$$

Since $\boldsymbol{\beta}$ does not depend on k, this requires $(\eta_{\text{eff}}(\mathbf{s}_{k_1}, \mathbf{r}) - 1) \cdot \phi_{k_1}^{\text{base}} = (\eta_{\text{eff}}(\mathbf{s}_{k_2}, \mathbf{r}) - 1) \cdot \phi_{k_2}^{\text{base}}$ for all k_1, k_2, which fails in general since both η_{eff} and ϕ_k^{base} vary across scatterers. Therefore, receiver-shared additive conditioning cannot represent the multiplicative structure.

Step 3: Real-valued scaling is insufficient. Consider a real-valued scaling model $\phi_k = \alpha \cdot \phi_k^{\text{base}}$ with $\alpha \in \mathbb{R}$. Component-wise:

$$\begin{bmatrix} \text{Re}(\phi_k^{(n)}) \\ \text{Im}(\phi_k^{(n)}) \end{bmatrix} = \begin{bmatrix} \alpha & 0 \\ 0 & \alpha \end{bmatrix} \begin{bmatrix} x_n \\ y_n \end{bmatrix}. \tag{38}$$

This is a diagonal matrix that can only scale amplitude, not rotate phase. Matching $\mathbf{M}(\eta_{\text{eff}})$ from Equation (36) requires $b = 0$, *i.e.*, $\text{Im}(\eta_{\text{eff}}) = 0$. From Equation (10), $\text{Im}(\eta_{\text{eff}}) = 0$ only when $\frac{2\pi}{\lambda}\|\mathbf{s}_k - \mathbf{r}\| = n\pi$ for integer n, which holds only at discrete distances. Therefore, real-valued scaling fails for general receiver positions.

Step 4: Complex affine modulation suffices. An element-wise affine modulation $(1+\boldsymbol{\alpha})\odot\phi_k^{\text{base}}+\boldsymbol{\beta}$ with $\boldsymbol{\alpha},\boldsymbol{\beta}\in\mathbb{C}^d$ parameterizes complex scale and shift for each component of ϕ_k^{base}. Setting $\boldsymbol{\alpha}=(\eta_{\text{eff}}-1)\cdot\mathbf{1}$ and $\boldsymbol{\beta}=\mathbf{0}$ recovers the exact product:

$$(1+\boldsymbol{\alpha})\odot\phi_k^{\text{base}}=\eta_{\text{eff}}\cdot\phi_k^{\text{base}}. \tag{39}$$

The additive term $\boldsymbol{\beta}$ provides additional capacity to absorb residual effects not captured by η_{eff} alone, such as multi-bounce interactions on the last segment. Since affine modulation succeeds where both receiver-shared additive conditioning (Step 2) and real-valued scaling (Step 3) fail, it preserves the multiplicative structure of the physics while providing slack for residual effects. □

C.3 Proof of Corollary 3

Proof. We show that the scatterer index k carries no information beyond the per-scatterer inputs (d_k, T_k) required to evaluate η_{eff}, and therefore separate per-scatterer networks cannot improve over a single shared network.

Step 1: Index-independence. From Equation (10), the effective propagation factor for any scatterer k is:

$$\eta_{\text{eff}}(\mathbf{s}_k,\mathbf{r})=f(d_k,T_k)=\frac{\lambda}{4\pi d_k}\cdot e^{-j\frac{2\pi}{\lambda}d_k}\cdot T_k, \tag{40}$$

where $d_k=\|\mathbf{s}_k-\mathbf{r}\|$ and $T_k=\prod_{m\in\mathcal{I}(k,\mathbf{r})}\tau_m$. The function f does not depend on the scatterer index k: for any two scatterers $k_1\neq k_2$, if $(d_{k_1},T_{k_1})=(d_{k_2},T_{k_2})$, then $\eta_{\text{eff}}(\mathbf{s}_{k_1},\mathbf{r})=\eta_{\text{eff}}(\mathbf{s}_{k_2},\mathbf{r})$. In the architectural implementation we augment the network inputs with the direction $\hat{\mathbf{v}}_k$ and mean density $\bar{\rho}_k$ to facilitate learning T_k from geometric cues, but these are not required by the theoretical form of f.

Step 2: Per-scatterer networks are redundant. Suppose we use separate networks $\{\hat{f}_k\}_{k=1}^K$, one per scatterer, each approximating f. Since f is index-independent from Step 1, the optimal solution for every $\hat{f}_k$ is identical:

$$\hat{f}_{k_1}^*=\hat{f}_{k_2}^*=\cdots=\hat{f}_K^*=f,\quad\forall\,k_1,k_2. \tag{41}$$

Therefore, at optimality, all per-scatterer networks converge to the same function, making the separate parameterizations redundant.

Step 3: Shared network suffices. Define a single shared network $\hat{f}$ that takes per-scatterer inputs for any k. From Step 1, $\hat{f}$ must approximate $f(d_k,T_k)$ over the union of all per-scatterer inputs:

$$\hat{f}\approx f\quad\text{on}\quad\bigcup_{k=1}^{K}\{(d_k,T_k):\mathbf{r}\in\mathbb{R}^3\}. \tag{42}$$

Since f is continuous for $d_k>0$ and the input domain is bounded in any physical scene, the universal approximation theorem guarantees that a single network $\hat{f}$ with sufficient capacity can approximate f to arbitrary precision. Therefore, a single shared network with per-scatterer inputs suffices for all k. □

D Single-Kernel Multi-Receiver Rasterization

The core mechanism behind RxGS's multi-receiver inference is a single CUDA call that renders all N receivers at once. This exploits the fact that the sort and per-Gaussian projection depend only on the transmitter, not on the receivers. For one transmitter at $\mathbf{t}$ serving N receivers $\{\mathbf{r}_j\}_{j=1}^N$, the pipeline runs three stages, summarized in Algorithm 1.

Transmitter-Side Preprocessing. We compute Gaussian projections, per-tile bounds, and the depth-sorted Gaussian list using only $\mathbf{t}$ and the Gaussian state. These results are receiver-independent and shared across all N rasterizations. This stage therefore runs once per transmitter regardless of receiver count.

Per-Receiver Directional Signal. We evaluate the FLE/SH directional basis on the transmitter-side directions once, then reduce against the per-receiver coefficient tensor of shape $(N,\,K,\,2,\,L)$ in a

Algorithm 1 Multi-Receiver Spectrum Rendering for One Transmitter.

Require: Gaussian state $\{\mathbf{p}_k,\ \mathbf{C}_k,\ \tau_k,\ \phi_k^{\text{base}}\}_{k=1}^{K}$
Require: Transmitter position $\mathbf{t}$, receiver positions $\{\mathbf{r}_j\}_{j=1}^{N}$
Require: Angular grid of size $H_\theta \times W_\varphi$ over elevation and azimuth
Ensure: Spectra $\left\{\hat{S}^{(j)} \in \mathbb{R}^{2\times H_\theta \times W_\varphi}\right\}_{j=1}^{N}$ (real and imaginary)

1: // Stage 1: Transmitter side, computed once
2: Project Gaussians onto the unit sphere centered at $\mathbf{t}$ and compute per-Gaussian direction $\hat{\mathbf{d}}_k$ and angular tile bounds
3: Build per-tile depth-sorted Gaussian list via one `cub::SortPairs` call over (tile, depth) keys
4: // Stage 2: Per-receiver coefficients, one batched call
5: $\left\{\phi_k^{(j)}\right\} \leftarrow$ `rxcond.forward_batched`$\left(\phi_k^{\text{base}},\ \{\mathbf{r}_j\},\ \{\mathbf{p}_k\}\right)$ ▷ shape $(N,\ K,\ 2,\ L)$
6: $\left\{s_k^{(j)}\right\} \leftarrow$ `eval_fle`$\left(B\left(\hat{\mathbf{d}}_k\right),\ \left\{\phi_k^{(j)}\right\}\right)$ ▷ basis B shared, reduce over L
7: // Stage 3: Single CUDA kernel with gridDim.$z = N$
8: **launch** `render_multirx<<<(tiles`$_\theta$`, tiles`$_\varphi$`, N), block>>>`
9: **for all** angular tile $(t_\theta,\ t_\varphi)$ and receiver $j \in [1..N]$ in parallel **do**
10: **for all** direction sample $(\theta,\ \varphi)$ in tile $(t_\theta,\ t_\varphi)$ in parallel **do**
11: $\mathcal{T} \leftarrow 1, C \leftarrow 0$
12: **for all** Gaussian k in tile's sorted list (front to back) **do**
13: $w_k \leftarrow \tau_k \cdot \exp\left(-\frac{1}{2}\boldsymbol{\Delta}_{k,\theta\varphi}^{\top}\mathbf{C}_k^{-1}\boldsymbol{\Delta}_{k,\theta\varphi}\right)$
14: $C \leftarrow C + \mathcal{T}\, w_k\, s_k^{(j)}$ ▷ per-receiver signal
15: $\mathcal{T} \leftarrow \mathcal{T}\,(1 - w_k)$
16: **if** $\mathcal{T} < \epsilon$ **then break**
17: **end if**
18: **end for**
19: $\hat{S}_{\theta\varphi}^{(j)} \leftarrow C$
20: **end for**
21: **end for**

single broadcasted operation. This yields the per-receiver signal tensor $\left\{s_k^{(j)}\right\} \in \mathbb{R}^{N\times K\times 2}$ in one batched call rather than N separate evaluations.

Multi-Receiver Rasterization. A single CUDA kernel is launched with grid dimension gridDim $=$ $(\text{tiles}_\theta,\ \text{tiles}_\varphi,\ N)$, placing the receiver index on the z-axis. Each thread block reads its receiver-specific signal slice and writes its receiver-specific output slot, while the shared transmitter-side state of projections, sorted list, and tile ranges is read by every block. The backward kernel runs with the same gridDim.$z = N$ to compute per-receiver $\partial\mathcal{L}/\partial\phi_k^{(j)}$ and accumulate Gaussian-level gradients across all (j, direction sample) threads via `atomicAdd`.

Shared Computation. The standard 3DGS rasterizer computes lines 2 and 3 once, then runs the per-direction alpha-blend once for a single spectrum. For multi-receiver rendering, lines 2 and 3 depend only on the transmitter-anchored sphere rays and are reused across receivers. The directional basis evaluated at $\hat{\mathbf{d}}_k$ also depends only on the transmitter and is computed once on line 6. The per-receiver computation reduces to the directional signal $s_k^{(j)}$, the FLE/SH coefficient sequence $\phi_k^{(j)}$ projected against the shared basis, producing one $(N,\ K,\ 2)$ signal tensor. The per-direction rendering then runs along an extra batch axis $j \in [1..N]$, mapped to blockIdx.z at kernel launch. Each receiver writes to its own output slot without atomic conflicts.

E Implementation Details

We implement RxGS in PyTorch, with custom CUDA kernels for RF signal tracing and rasterization to support the differentiable RF rendering pipeline. All experiments are conducted on a single NVIDIA H100 GPU. This section describes the implementation of the FLE-based scene representation. Other representation variants (SH) and signal modalities (RSSI, CSI) follow the same training pipeline and optimization strategy. Dataset-specific hyperparameters are reported in Table 5.

E.1 Scene Representation

The RF environment is represented as a set of K 3D Gaussians. Each Gaussian stores five attributes: position $\mathbf{p} \in \mathbb{R}^3$, anisotropic scale $\mathbf{s} \in \mathbb{R}^3$ (log-space), rotation quaternion $\mathbf{q} \in \mathbb{R}^4$, sigmoid-activated transmittance $\tau \in [0, 1]$, and FLE coefficients $\mathbf{f} \in \mathbb{R}^{L \times C}$. Here $L = (\ell_{\max} + 1)^2$ is the number of FLE basis functions (matching the radiance-component count from the main text), and C is the number of signal channels. The storage $\mathbf{f}$ realizes the base coefficients ϕ_k^{base} from §4.3 in the implementation. The covariance matrix is constructed as $\mathbf{\Sigma} = \mathbf{R}\mathbf{S}\mathbf{S}^\top\mathbf{R}^\top$, where $\mathbf{R}$ and $\mathbf{S}$ are the rotation and scaling matrices built from $\mathbf{q}$ and $\mathbf{s}$.

Initialization. We initialize up to $50{,}000$ Gaussians from a point cloud sampled from the scene. Scales are set to the log of nearest-neighbor distances, computed via a CUDA KNN module. Rotations use identity quaternions, transmittance starts at $\sigma^{-1}(0.1)$, and FLE coefficients at zero.

Table 5: Hyperparameter settings for RxGS across all datasets.

Parameter	**BLE RSSI**	**RFID Spectrum**	**WiFi CSI**
Initial Gaussians (K_{init})	50,000	50,000	30,000
Max FLE degree ($\ell_{\max}$)	9	9	4
FLE ramp interval (T_{ramp})	500	500	100
Ray directions ($n_\varphi \times n_\theta$)	36×9	360×90	72×18
Sphere radius (r_{rx})	4.22	1.00	1.00
Position LR (initial)	5.05×10^{-5}	1.60×10^{-4}	2.78×10^{-4}
Position LR (final)	2.06×10^{-6}	1.60×10^{-6}	1.60×10^{-6}
Feature LR	1.14×10^{-3}	5.00×10^{-3}	1.53×10^{-2}
Rest LR ratio	0.16	0.20	0.80
Transmittance LR	3.79×10^{-2}	1.00×10^{-2}	1.27×10^{-3}
Scaling LR	2.90×10^{-3}	5.00×10^{-3}	5.00×10^{-3}
Rotation LR	5.02×10^{-4}	1.00×10^{-3}	1.00×10^{-3}
Densification threshold (τ_g)	1.97×10^{-4}	2.00×10^{-4}	1.88×10^{-4}
Transmittance reset interval	4,000	3,000	2,000
Fourier bands (F)	6	6	5
MLP hidden dim (d)	64	64	256
Component embedding dim (d_c)	16	16	16
Occupancy probe samples (S)	16	16	16
Grid resolution	128^3	128^3	128^3
Stage I iterations	30K	30K	30K
Stage II iterations	100K	60K	100K

E.2 Signal Rendering Pipeline

For each Gaussian, the renderer computes a complex signal pair from its directional FLE coefficients along the transmitter-to-Gaussian direction. For a direction $\hat{\mathbf{d}} = (\theta, \varphi)$ from the transmitter to the Gaussian center, the real and imaginary components of the radiated signal are:

$$R_e = \sum_{l=0}^{\ell_{\max}} \sum_{m=-l}^{l} \left[a_{lm} \cos(m\varphi) - b_{lm} \sin(m\varphi)\right] N_{lm} P_l^{|m|}(\cos\theta) \tag{43}$$

$$R_i = \sum_{l=0}^{\ell_{\max}} \sum_{m=-l}^{l} \left[a_{lm} \sin(m\varphi) + b_{lm} \cos(m\varphi)\right] N_{lm} P_l^{|m|}(\cos\theta) \tag{44}$$

where $a_{lm}, b_{lm} \in \mathbf{f}$ are learnable coefficients, $P_l^{|m|}$ are associated Legendre polynomials computed via batched three-term recurrence, and $N_{lm} = \sqrt{\frac{(2l+1)}{4\pi} \cdot \frac{(l-|m|)!}{(l+|m|)!}}$ is the normalization factor. The

custom CUDA rasterizer detailed in Appendix D combines these signals with each Gaussian's covariance and transmittance through covariance-weighted accumulation. The final output amplitude $|R| = \sqrt{R_e^2 + R_i^2 + \epsilon}$ uses a numerical stabilizer $\epsilon = 10^{-8}$. The rendering grid uses a sphere of radius r_{rx} centered at the transmitter.

E.3 Loss Function

Both training stages use a modality-specific loss: L_1 for scalar RSSI, L_2 for complex-valued CSI, and a composite loss for spatial spectrum. For a rendered spectrum $\hat{\mathbf{y}}$ and ground-truth spectrum $\mathbf{y}$, the composite loss is:

$$\mathcal{L} = (1 - \lambda_{\text{ssim}} - \lambda_{\text{fft}}) \|\hat{\mathbf{y}} - \mathbf{y}\|_1 + \lambda_{\text{ssim}} (1 - \text{SSIM}(\hat{\mathbf{y}}, \mathbf{y})) + \lambda_{\text{fft}} \|\mathcal{F}(\hat{\mathbf{y}}) - \mathcal{F}(\mathbf{y})\|_2^2 \quad (45)$$

Here λ_{ssim} and λ_{fft} weight the structural-similarity and Fourier-domain terms, respectively. The operator $\mathcal{F}$ denotes the 2D orthonormal FFT, and SSIM uses an 11×11 Gaussian window.

E.4 Stage I: Geometry Learning

Stage I trains all Gaussian parameters using measurements from a single reference receiver. At each iteration, a random training sample from the reference receiver is selected, and the rendered signal is compared with the measurement using the modality-specific loss.

Optimizer. We use Adam with per-parameter learning rates listed in Table 5. The position LR decays exponentially from its initial to final value with a delay multiplier of 0.01. Higher-degree FLE coefficients are additionally scaled by 0.2 to prioritize low-frequency geometry early in training. The FLE degree is progressively increased from 0, incrementing every T_{ramp} iterations up to $\ell_{\max}$. Stage I runs for 30K iterations total.

Adaptive Densification. Gaussians are densified based on accumulated positional gradients every 100 iterations, from iteration 500 to the stage midpoint (iteration 15K). Gaussians exceeding the densification threshold τ_g listed in Table 5 are cloned if their scale is below the scene extent threshold, or split into two children with scales reduced by factor 0.8 if above. Gaussians with projected screen-space radius above a pixel threshold, or world-space extent exceeding $0.1\times$ the scene extent, are pruned. To avoid saturation, transmittance resets to $\sigma^{-1}(0.01)$ as specified in Table 5.

E.5 Stage II: Receiver-Conditioned Radiance

With geometry frozen, Stage II trains the FLE coefficients and the conditioning modules across all receivers. At each iteration, a random training sample from an available receiver is selected. The conditioning modules then modulate the base FLE coefficients using its position, and the modality-specific loss compares the rendered signal with the corresponding measurement.

Global Branch. Table 5 specifies F frequency bands per spatial dimension for the learnable Fourier encoding of receiver position. The bands start at 2^k spacing and are optimized during training. For each FLE component, a 3-layer ReLU MLP with hidden dimension d takes the concatenation of the encoded RX position, per-component degree and order features, and a learnable per-component embedding of dimension d_c as input. The MLP produces per-component scale and shift parameters that modulate the base FLE coefficients. This modulation is computed once per component and broadcast to all K Gaussians, giving $\mathcal{O}(L)$ cost independent of Gaussian count.

Local Branch. At the start of Stage II, a 3D occupancy grid of resolution 128^3 is built by splatting each frozen Gaussian into a voxel density field using its position, transmittance, and scale with a 2σ radius. For each Gaussian k, we cast a ray toward the receiver and probe the grid at S uniformly spaced points (from $t = 0.05$ to $t = 0.95$) via nearest-neighbor lookup, computing path transmittance $T_k = \prod_{s=1}^{S}(1 - \rho_s)$ and mean density $\bar{\rho}_k$. A shared MLP takes $[\hat{\mathbf{v}}_k,\ d_k,\ T_k,\ \bar{\rho}_k] \in \mathbb{R}^6$ as input and produces per-Gaussian scale and shift parameters. Both MLP final layers are zero-initialized, making the initial conditioning transform the identity.

Optimizer. FLE coefficients and conditioning modules are trained with separate Adam optimizers, with learning rates listed in Table 5. Higher-degree FLE coefficients are scaled by 0.2 as in Stage I.

F Dataset Details

All three datasets use random seed 8371 for reproducible train/test splits. TX denotes transmitter and RX denotes receiver throughout this section.

F.1 BLE RSSI

The BLE RSSI dataset is collected in a real-world nursing home occupying $\sim$15,000 ft^2 [5]. $N_{\mathrm{RX}} = 21$ BLE gateways operating at 2.4 GHz are deployed throughout the facility, each measuring the received signal strength (RSSI in dBm) from a moving BLE node. TX positions are collected by random walks using 30 BLE nodes, yielding $N_{\mathrm{TX}} = 6{,}000$ TX positions in total. Each sample is a 21-dimensional tuple of RSSI readings from the 21 receivers. Readings below -100 dBm are excluded. A random 80/20 split yields 4,800 training and 1,200 test positions.

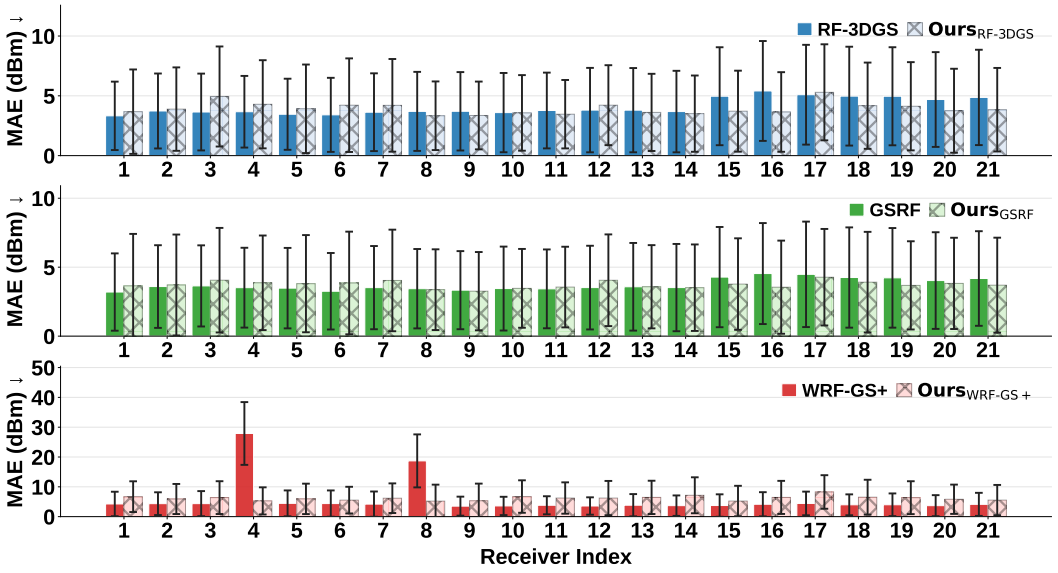


Figure 4: Per-receiver MAE breakdown on BLE RSSI for each baseline and its receiver-conditioned counterpart (lower is better). Adding our receiver conditioning produces more uniform performance across receivers and eliminates the catastrophic outliers seen in WRF-GS+ at receivers 4 and 8.

F.2 RFID Spatial Spectrum

Collecting real-world spatial spectrum data requires a calibrated antenna array and precise TX positioning across hundreds of positions, making large-scale data collection prohibitively expensive. Furthermore, no public multi-receiver spatial spectrum dataset currently exists. We therefore generate this dataset via simulation using the NVIDIA Sionna ray tracer [45, 46]. Sionna is a physics-based, differentiable ray tracing engine that models line-of-sight, specular reflection, diffuse scattering, and refraction, with material properties calibrated to ITU-R standards. Its GPU acceleration enables large-scale dataset generation at simulation speeds.

The scene is a cluttered conference room (8 m $\times$ 6 m $\times$ 3 m), as shown in Figure 5. Each measurement is a 90×360 grayscale image capturing the angular power spectrum over elevation and azimuth at 2.4 GHz. The dataset contains $N_{\mathrm{TX}} = 5{,}089$ TX positions and $N_{\mathrm{RX}} = 21$ distributed RX positions at $z = 2$ m height, yielding $N_{\mathrm{TX}} \times N_{\mathrm{RX}} = 106{,}869$ spectrum images in total. We use an 80/20 random split (4,071 train / 1,018 test TX positions).

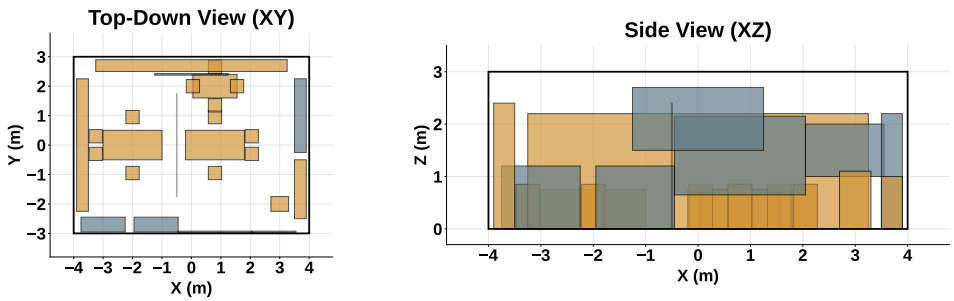


Figure 5: Floor plan of the conference room (8 m × 6 m × 3 m) used for the RFID spectrum dataset.

Scene Construction. Each scene is modeled as a rectangular room with six surfaces (floor, ceiling, four walls), built as Mitsuba triangle meshes. Furniture items including tables, chairs, bookshelves, metal cabinets, and glass partitions are represented as axis-aligned box meshes. Each object is assigned an ITU radio material (*e.g.*, concrete, plasterboard, wood, metal, glass) with calibrated relative permittivity, conductivity, and scattering coefficients following the ITU-R P.2040 standard.

Channel Simulation. For each TX–RX pair, we place a single isotropic TX antenna and, at the RX, an 8×8 uniform planar array (UPA) with half-wavelength spacing at $f = 2.4\,\text{GHz}$. Sionna's `PathSolver` performs ray tracing with up to 5 bounces, including line-of-sight, specular reflection, diffuse reflection, and refraction. The solver returns $\mathbf{H} \in \mathbb{C}^{N_{\text{ant}} \times N_{\text{sub}}}$ with $N_{\text{ant}} = 64$ antennas and $N_{\text{sub}} = 512$ subcarriers across 100 MHz.

Spectrum Computation. We compute the angular power spectrum via conventional beamforming (CBF). A 2D Hanning window is applied across the array to suppress sidelobes. Per-element phase and amplitude errors ($\pm 15^\circ$, ± 1.5 dB) are introduced for realism. The spatial covariance matrix is estimated as $\mathbf{R}_{\text{spec}} = \mathbf{H}\mathbf{H}^H / N_{\text{sub}}$. For each scan direction (φ, θ) on a 1°-resolution grid covering azimuth $[0^\circ, 360^\circ)$ and elevation $[0^\circ, 90^\circ)$, the beamforming power is:

$$P(\varphi, \theta) = \mathbf{a}^H(\varphi, \theta)\, \mathbf{R}_{\text{spec}}\, \mathbf{a}(\varphi, \theta) \tag{46}$$

where $\mathbf{a}(\varphi, \theta) = \exp\big(j\, k\, \mathbf{p}_{\text{ant}}^\top \hat{\mathbf{u}}(\varphi, \theta)\big) \in \mathbb{C}^{N_{\text{ant}}}$ is the steering vector, $k = 2\pi/\lambda$ is the wavenumber, $\mathbf{p}_{\text{ant}}$ contains the antenna element positions, and $\hat{\mathbf{u}}$ is the unit direction vector. The 3GPP TR 38.901 element radiation pattern is applied in the array's local coordinate frame. The resulting spectrum is peak-normalized to 0 dB and saved as a 90×360 grayscale image.

TX/RX Placement. We place $N_{\text{RX}} = 21$ RX positions at uniformly random (x, y) positions within the room with a minimum inter-RX spacing constraint. TX positions are sampled randomly within the room volume and validated against furniture collisions via rejection sampling.

F.3 WiFi CSI

Prior work [5, 19] uses the Argos dataset [49] for CSI evaluation, where all antennas are co-located at a single position. For multi-receiver evaluation, we use the KU Leuven Ultra-Dense Indoor MaMIMO CSI Dataset [47], collected using a 64-antenna massive MIMO testbed operating at a center frequency of 2.61 GHz with 20 MHz bandwidth. The original measurements comprise 252,004 channel snapshots over a $1.25\,\text{m} \times 1.25\,\text{m}$ transmitter grid at 5 mm spatial resolution, with 100 complex subcarriers per antenna. In the distributed line-of-sight scenario, eight groups of eight antennas surround a $\sim 5\,\text{m} \times 5\,\text{m}$ room in an octagonal layout at $z = 1\,\text{m}$.

Preprocessing. We apply five preprocessing steps to the raw data. First, phase calibration removes the random carrier phase offset per antenna, ensuring coherent combination across elements. Second, the 8 antennas within each group are coherently averaged after calibration, yielding one effective RX per group and $N_{\text{RX}} = 8$ distributed RXs in total. Third, delay-domain denoising truncates the channel impulse response to the first 10 of 100 taps via IFFT, truncation, and FFT, suppressing noise-dominated late arrivals that carry no useful propagation information. Fourth, the 100 subcarriers are averaged into $N_{\text{sc}} = 26$ frequency bins to reduce redundancy while preserving the frequency-selective fading structure. Fifth, nearby TX positions are spatially averaged into $N_{\text{TX}} = 6{,}000$ output positions to reduce wavelength-scale phase variation and produce a spatially consistent dataset suitable for scene-level learning.

Final Dataset. The resulting dataset contains $N_{\mathrm{TX}} = 6{,}000$ TX positions and $N_{\mathrm{RX}} = 8$ distributed RX positions, yielding $N_{\mathrm{TX}} \times N_{\mathrm{RX}} = 48{,}000$ complex CSI vectors of dimension 52 ($N_{\mathrm{sc}} = 26$ subcarriers $\times$ 2 for real and imaginary), normalized by the global maximum magnitude. We use an 80/20 random split (4,800 train / 1,200 test TX positions).

G Additional Experiments

This appendix presents additional experiments that complement the main text. Appendix G.1 breaks down Table 1 into per-receiver results. Appendix G.2 visualizes per-subcarrier CSI traces. Appendix G.3 expands Table 2 into per-backbone training, inference, and storage breakdowns. Appendix G.4 shows spectrum visualizations on seen and unseen receivers. Appendix G.5 analyzes how unseen-receiver accuracy scales with training receivers. Appendix G.6 measures sensitivity to the Stage I reference receiver. Appendix G.7 reports a network planning study. Appendix G.8 examines receiver-level accuracy gaps and their geometric correlates. Appendix G.9 compares line-of-sight and non-line-of-sight pairs and assesses the occlusion probe.

G.1 Per-Receiver Performance Breakdown

Table 1 reports metrics averaged across all receivers, which can hide substantial per-receiver variation. Per-receiver baselines fit each receiver in isolation, and aggregate metrics can obscure high-error receivers. We therefore break the results down receiver-by-receiver on all three datasets, pairing each baseline with its receiver-conditioned counterpart that augments the same backbone with our conditioning module as a single shared model.

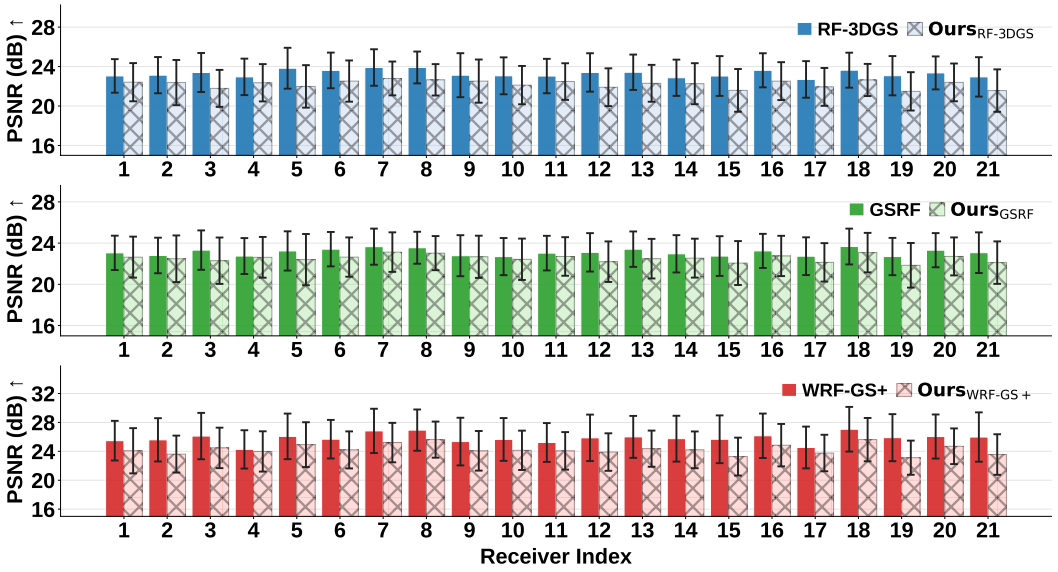


Figure 6: Per-receiver PSNR breakdown on RFID spatial spectrum for each baseline and its receiver-conditioned counterpart (higher is better). One shared model covers all 21 receivers. Table 1 summarizes the corresponding average PSNR.

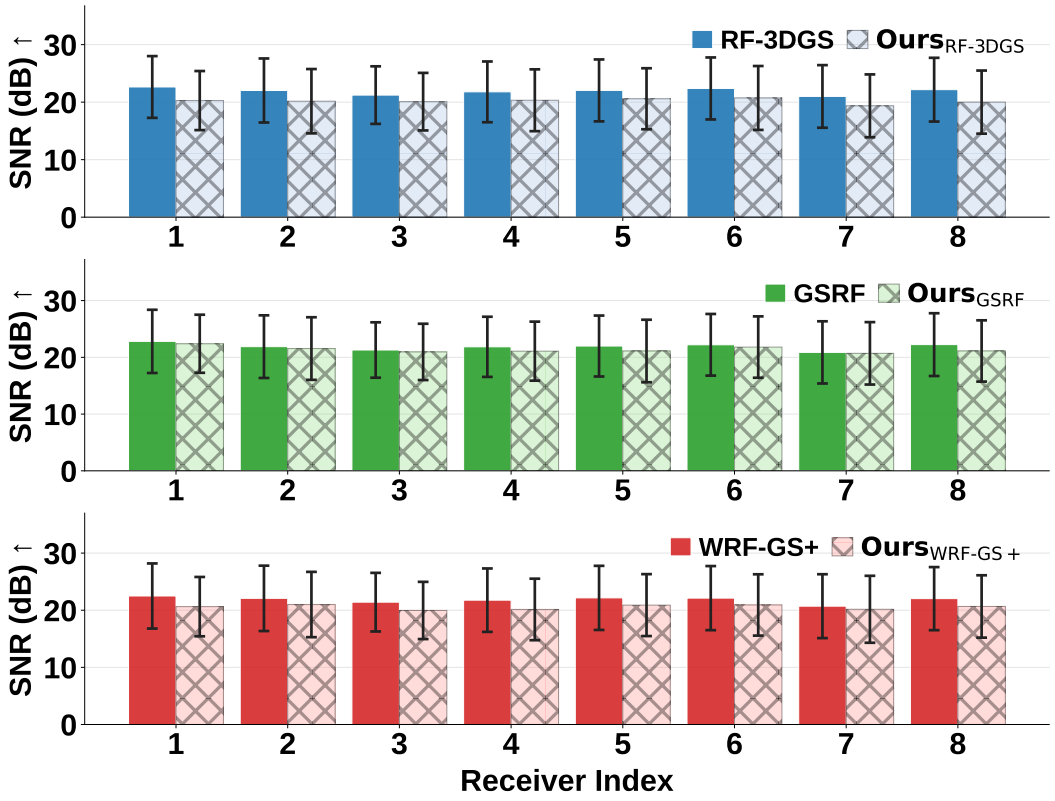


Figure 7: Per-receiver SNR breakdown on WiFi CSI for each baseline and its receiver-conditioned counterpart (higher is better). One shared model covers all 8 distributed receivers. Table 1 summarizes the corresponding average SNR.

Figures 4–7 compare all receivers across three baseline backbones, RF-3DGS [18], GSRF [20], and WRF-GS+ [19], and their unified counterparts on BLE RSSI, RFID spatial spectrum, and WiFi CSI respectively. Four patterns emerge.

Uniformity across receivers. For all three backbones and all three datasets, the receiver-conditioned variant yields a flatter metric profile across the receiver index. This matches the behavior of a model that learns a transferable receiver-conditioning function rather than memorizing per-receiver mappings. Weakly-covered receivers benefit from data collected at others.

Outlier removal. On BLE RSSI, per-receiver WRF-GS+ has two high-error receivers: receiver 4 at about 28 dBm and receiver 8 at about 19 dBm MAE, both over 5× its median error. The aggregate mean obscures their contribution. Receiver-conditioned WRF-GS+ removes both outliers.

Spectrum and CSI Accuracy. Table 1 shows that the shared model generally tracks the per-receiver profile on RFID spectrum and WiFi CSI. Average PSNR and SNR are lower than those of specialized models by 0.8 to 1.7 dB.

Behavior Across Backbones. The conditioning design applies to all three tested 3DGS backbones, but its accuracy effect depends on the backbone and signal modality. It improves the BLE mean for RF-3DGS and GSRF, removes the isolated WRF-GS+ failures while slightly raising its mean, and trades spectrum and CSI accuracy for one shared model and lower deployment cost.

These observations corroborate the aggregated numbers in Table 1.

G.2 Qualitative CSI Trace Visualization

Setting. We complement the aggregate SNR numbers in the main text with per-subcarrier CSI traces produced by our model on the WiFi CSI dataset. We plot predicted and ground-truth CSI vectors H_{Pred}, $H_{\mathrm{GT}} \in \mathbb{C}^{26}$ at three randomly selected test transmitters. The top row of Figure 8 shows amplitude $|H(f)|$ normalized per panel by $\max(|H_{\mathrm{GT}}|, |H_{\mathrm{Pred}}|)$. The bottom row shows phase wrapped to $[0, 2\pi)$.

Results. Across all three TX positions, the predicted amplitude closely follows the ground-truth frequency response. Each panel reproduces the multipath-induced amplitude peaks and nulls at the correct subcarrier indices. The overall envelope is preserved with only minor shifts of ± 1 to 2 subcarriers near sharp nulls. The phase traces track the ground-truth dispersion equally well. The 2π wraparound visible in TX1 occurs at matching subcarrier indices for prediction and reference. No panel shows a systematic bias or catastrophic drift. This indicates that RxGS captures both the magnitude and phase structure of the channel, not only amplitude statistics.

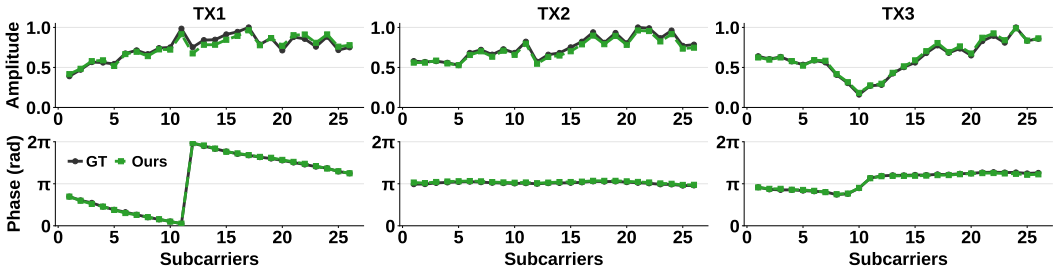


Figure 8: Qualitative CSI traces on the WiFi CSI dataset at three transmitter (TX) positions. Top row: amplitude $|H(f)|$ normalized per panel. Bottom row: phase wrapped to $[0, 2\pi)$. RxGS (green, dashed) tracks the ground truth (black, solid) in both amplitude and phase.

G.3 Computational Cost: Detailed Breakdown

We expand on Table 2 and explain why the gains differ across backbones.

Implicit versus explicit scene representations. Table 2 shows different storage and compute profiles for implicit and explicit models. NeRF^2 stores only 2.6 MB but takes 14.3 h to train and 2436 ms to render $N = 21$ spectra. GSRF + RxGS stores 26 MB yet trains in 0.6 h and renders in 31 ms. The two paradigms swap their cost profiles:

- **Implicit methods (NeRF^2, GRaF)** encode the entire scene as a continuous function of position and direction using a small MLP. Each transmitter-receiver sample casts one ray per pixel and integrates by querying the MLP at $\mathcal{O}(64\text{–}256)$ depth points, totaling roughly 2 M MLP passes for a single 90×360 spectrum, with the same number of backward passes during training. Storage is small (only MLP weights), but compute scales as $\mathcal{O}(\text{rays} \cdot \text{depth_samples} \cdot \text{MLP_FLOPs})$ and grows linearly with output resolution.
- **Explicit methods (RF-3DGS, GSRF, WRF-GS+)** parameterize the scene as 10k to 40k Gaussian primitives, each carrying position, scale, rotation, opacity, and a directional coefficient vector with $L = 16$ to 100 entries. Storage scales as $\mathcal{O}(K \cdot L)$ and dominates total bytes, but the CUDA rasterizer projects all Gaussians once, sorts them per tile, and runs a single per-pixel alpha-blend pass with $\mathcal{O}(\text{Gaussians overlapping the tile})$ work, typically tens of Gaussians per tile. The pipeline is memory-bandwidth bound and parallelizes well on a GPU. For one transmitter, the RxGS variants synthesize signals at all $N = 21$ receivers in 31 to 43 ms.

Implicit methods use less storage but require more computation per sample, whereas explicit methods use more storage with lower per-sample computation.

Training. RxGS totals 90k iterations per backbone: 30k for Stage I geometry pre-training and 60k for Stage II multi-receiver fitting. The per-receiver baselines run N independent fits of T_{base} iterations each, with $T_{\text{base}} = 200$k for WRF-GS+ and 30k for GSRF and RF-3DGS. The raw iteration ratio is $N \cdot T_{\text{base}}/90$k, with measured speedups also reflecting per-iteration cost differences.

- **RF-3DGS:** $2.8 \rightarrow 0.4$ h, a $7\times$ speedup that matches the raw iteration ratio of $21 \cdot 30\text{k}/90\text{k} = 7\times$ almost exactly, with similar per-iteration cost for Stage II and per-receiver training.
- **GSRF:** $5.5 \rightarrow 0.6$ h, a $9\times$ speedup against the same $7\times$ raw ratio. The extra factor comes from Stage II running per-iteration slightly faster than per-receiver GSRF, which we attribute to amortizing FLE-basis evaluation across the batched receiver dimension.
- **WRF-GS+:** $13.6 \rightarrow 0.3$ h, a $45\times$ speedup against a raw iteration ratio of $21 \cdot 200\text{k}/90\text{k} \approx 47\times$. Multi-receiver overhead explains the roughly 4% difference. The main gain comes from one 90k-iteration run instead of 21 separate 200k-iteration fits.

NeRF^2 and GRaF are already shared models without N-way training. Their 14 to 16 h costs reflect roughly 600k iterations of volumetric sampling and backpropagation, consistent with the implicit paradigm above.

Inference. For the GSRF and RF-3DGS backbones, the multi-receiver rasterizer detailed in Appendix D renders all N receivers in one CUDA call. The per-method speedup over the per-receiver baseline depends on how much of the original cost was sharable versus per-pixel work:

- **GSRF:** $238 \rightarrow 31$ ms, a $7.6\times$ speedup that is the largest of the three because FLE-degree 9 with $L = 100$ components per Gaussian makes basis evaluation the dominant per-receiver cost. Our batched `eval_fle` now performs this evaluation once per transmitter and reuses it across all N receivers, eliminating the most expensive sharable component.
- **RF-3DGS:** $156 \rightarrow 35$ ms, a $4.4\times$ speedup. At SH degree 4 with $L = 25$, basis evaluation is less costly, and most savings arise from shared preprocessing and tile sorting rather than the per-pixel kernel.
- **WRF-GS+:** $52 \rightarrow 43$ ms, only a $1.2\times$ speedup. In the measured implementation, the receiver-independent DeformModel step is shared, whereas Gaussian rasterization runs separately for each receiver in Python. A native multi-receiver rasterizer offers a route to additional inference savings for this backbone.

The cost breakdown also explains two ordering effects. First, GSRF + RxGS ends up faster than RF-3DGS + RxGS at 31 versus 35 ms, despite per-receiver GSRF being slower than per-receiver RF-3DGS at 11.3 versus 7.45 ms. Once basis evaluation is amortized, the rasterizer dominates, and rasterizer cost scales with the number of Gaussians, of which RF-3DGS has more at 37.7k versus 31.8k. Second, NeRF^2 and GRaF have no batchable structure across receivers. Their per-transmitter cost remains N times the per-sample cost, yielding the 60 to $1900\times$ differences reported in the table.

Storage. Per-receiver baselines store N independent Gaussian-state checkpoints, whereas RxGS stores one. The deployable footprint decreases by roughly a factor of N:

- **GSRF:** $548 \rightarrow 26$ MB, a $21\times$ reduction. The absolute RxGS size is the largest of the three because GSRF carries $K = 32$k Gaussians at $L = 100$ FLE components, and storage scales as $K \cdot L \cdot C \cdot 4$ bytes plus roughly $10K$ bytes for geometry attributes.
- **RF-3DGS:** $182 \rightarrow 8.8$ MB, a $21\times$ reduction. Although RF-3DGS has more Gaussians than GSRF ($K = 38$k), its smaller $L = 25$ yields a smaller model.
- **WRF-GS+:** $75.6 \rightarrow 3.2$ MB, a $24\times$ reduction. The smallest absolute size comes from $K = 14$k at $L = 16$, plus a 2.2 MB DeformModel. The slightly higher reduction ratio reflects the same N collapse at smaller absolute scale.

The $3.2 \rightarrow 8.8 \rightarrow 26$ MB ordering of the RxGS rows therefore tracks $K \cdot L$ rather than the per-receiver baseline ranking. NeRF^2 and GRaF are shared models that do not multiply with N but pay this saving back in inference compute. Their sizes differ substantially (2.6 MB for NeRF^2 versus 143.7 MB for GRaF), because GRaF stores training-receiver reference spectra.

Receiver-count scaling. Per-receiver baselines require N separately trained and stored models. The shared approach stores one model and uses a batched rasterization call for the FLE and SH backbones. Stage II training processes receiver-dependent samples, batched inference writes N outputs and performs some receiver-dependent work, and the WRF-GS+ variant retains a render loop. Model storage is shared, while training and inference costs vary with receiver count and backbone. Table 2 measures the resulting savings at $N = 21$.

Receiver-Conditioned Scene MLPs. NeRF^2 [5] conditions a scene MLP on the receiver, while GRaF [33] conditions on both endpoints. This approach incurs per-ray volumetric sampling and MLP evaluation at each query. At the measured $N = 21$ setting, Table 2 shows that NeRF^2 and GRaF take roughly 60 to 1900$\times$ longer per transmitter than the shared 3DGS variants. RxGS retains the explicit 3DGS scene representation and applies receiver-conditioned modulation around frozen geometry. The global and local branches require $\mathcal{O}(L + K)$ network evaluations per receiver query, consistent with §4.3. The rasterizer in Appendix D shares transmitter-side projection and sorting across receivers. The two-stage decomposition trains geometry once and restricts receiver dependence to radiance.

G.4 Qualitative Spectrum Visualization

Figure 9 visualizes spectrum predictions on one training receiver (*RX1*) and two receivers excluded from training (*RX2*, *RX3*), each evaluated at three test transmitters (*TX1*, *TX2*, *TX3*). For visual clarity we show one representative per design family: RxGS for Gaussian splatting with receiver-position conditioning, NeRF^2 for continuous neural fields with receiver conditioning, and WRF-GS+ for per-receiver Gaussian splatting without conditioning. GSRF and RF-3DGS share the per-receiver structure of WRF-GS+ and exhibit similar failures at unseen receivers.

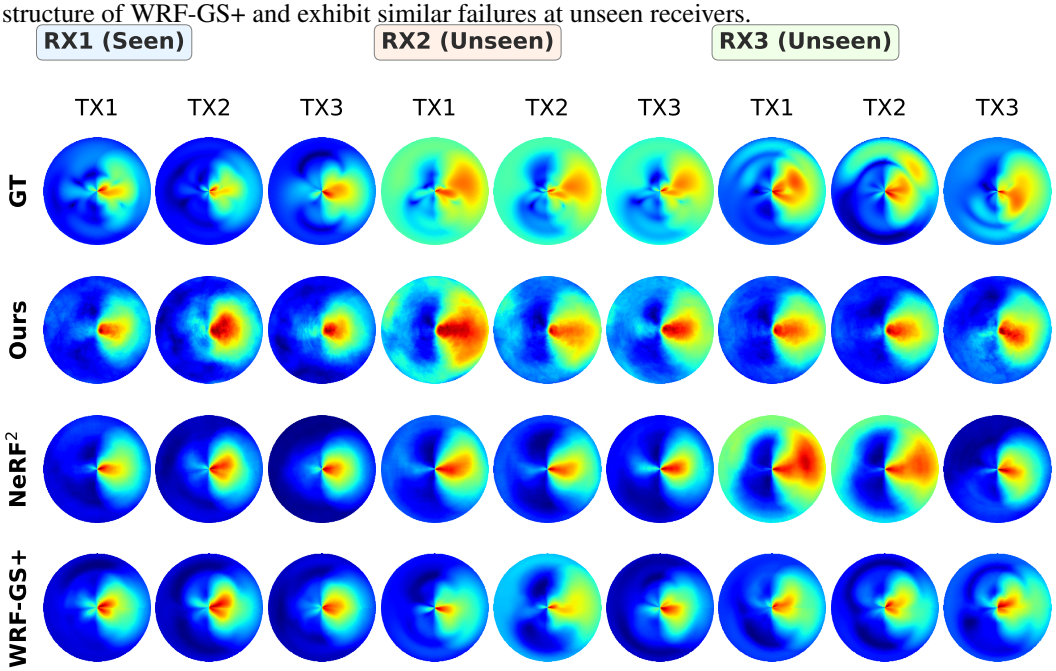


Figure 9: Qualitative comparison of RFID spatial spectrum prediction. *RX1* is seen during training, while *RX2* and *RX3* are excluded from training. RxGS produces the closest match to ground truth on unseen receivers. NeRF^2 predictions are overly smooth, missing sharp directional peaks. At receivers excluded from training, WRF-GS+ uses the closest trained-receiver checkpoint, yielding degraded predictions.

All methods recover the dominant structure on the seen receiver *RX1*. Prediction differences are more pronounced at the unseen receivers *RX2* and *RX3*.

RxGS Preserves Sharp Directional Peaks. On *RX2* and *RX3*, RxGS reproduces the narrow, bright lobes of the ground truth (*e.g.*, *RX2–TX1*, *RX2–TX2*, *RX3–TX1*) at the correct azimuth and elevation. Receiver conditioning transfers these peaks to unseen receivers.

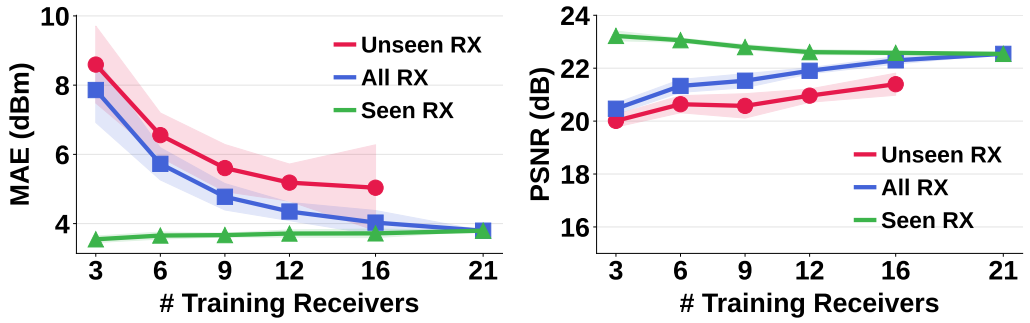
NeRF^2 Over-Smooths. NeRF^2 predictions on unseen receivers are plausible but visibly blurred. Peaks smear into broader, lower-amplitude regions, and fine directional structure is lost. The continuous MLP interpolates between training receivers, while the high-frequency lobes that distinguish their views are attenuated in these examples.

WRF-GS+ at Unseen Receivers. WRF-GS+ trains one model per receiver and lacks a query mechanism for new receiver positions. At receiver positions excluded from training, it uses the nearest trained-receiver checkpoint. The *RX2* and *RX3* predictions resemble a perturbed version of a different receiver's spectrum. The coarse shape is sometimes close, but peak positions and amplitudes are systematically wrong.

G.5 Scaling with Number of Receivers

Setting. We measure how generalization to unseen receivers depends on receiver diversity seen during training. For each $N \in \{3, 6, 9, 12, 16, 21\}$, we train a single unified RxGS model on N training receivers and evaluate on all 21 receivers, separating results into *seen* and *unseen* groups. To control subset-selection variance, we draw 7 random training subsets at each N and report mean and standard deviation. The $N{=}21$ point serves as an upper bound where all receivers are seen. Architecture, hyperparameters, and the transmitter train/test split remain fixed across runs. Only the number of training receivers varies.

Seen-Receiver Accuracy Is Flat in N. The seen-receiver curves in Figure 10 stay flat as the training-receiver count grows. On BLE, seen MAE varies only between 3.55 and 3.80 dBm despite the training set growing 7×. On spectrum, seen PSNR ranges from 22.58 to 23.22 dB across the same receiver-count settings. Across the tested subsets, shared geometry and receiver conditioning maintain similar seen-receiver accuracy as the number of jointly modeled receivers increases.



(a) BLE RSSI (b) RFID Spectrum

Figure 10: Scaling with the number of training receivers. Seen accuracy stays flat while unseen accuracy improves with N.

Unseen-Receiver Accuracy Improves with N. The unseen-receiver curves in Figure 10 drop substantially as N increases. On BLE, unseen MAE drops from 8.59 dBm at $N{=}3$ to 5.04 dBm at $N{=}16$, a 41% relative reduction. The seen-to-unseen gap shrinks from 5.04 to 1.32 dBm over the same range. On spectrum, unseen PSNR climbs from 20.01 dB at $N{=}3$ to 21.39 dB at $N{=}16$, narrowing the gap from 3.21 to 1.19 dB. Both curves converge to the seen-receiver value as $N \to 21$.

Implication. The combined pattern of flat seen accuracy and improving unseen accuracy indicates that the conditioning module learns a transferable, position-aware function of the receiver rather than memorizing per-receiver corrections. A small number of training receivers leaves the conditioning manifold too sparse to disambiguate receiver-driven variation from scene-driven structure, producing large extrapolation errors. Adding receivers densifies this manifold, enabling accurate interpolation to unseen positions. Diminishing returns beyond $N{=}12$ indicate that around a dozen training receivers cover much of the measured variation in the evaluated scenes. For practical deployment, this means one can train on a modest number of representative anchor receivers rather than collecting calibration data at every possible position.

G.6 Sensitivity to Reference Receiver Choice

Setting. Stage I requires a reference receiver to learn the scene geometry. Deployment of RxGS requires selecting a reference receiver for Stage I. We therefore evaluate the sensitivity of final performance to this selection. We compare 21 single-receiver reference choices against an averaged-signal initialization that pools measurements across all receivers, training the full pipeline on the BLE dataset under each choice. Stage II is trained across all 21 receivers with the same hyperparameters in every run, and we report per-receiver MAE averaged across all 21 test receivers.

Results. Figure 11 reports the resulting per-receiver MAE. The single-reference variants yield 3.94 ± 0.13 dBm across a range of $[3.67, 4.22]$ dBm, a 0.55 dBm spread or about 15% of the best single-reference MAE. The narrow spread across the 21 tested choices is consistent with the receiver-independent geometry representation derived in Appendix A. The averaged-signal initialization performs slightly better at 3.57 dBm, since pooling across receivers averages out per-receiver measurement noise and yields a cleaner geometry-learning signal. We therefore adopt the averaged-signal initialization as the default for all main-paper experiments. Each single-reference run learns Stage I geometry using only one receiver without signal averaging, and its MAE is 0.10 to 0.65 dBm above the 3.57 dBm averaged default. Averaging is a refinement rather than a requirement and uses only measurements already available to Stage II. In practice, however, either choice would yield comparable performance, and a practitioner can use whichever is most convenient for their deployment.

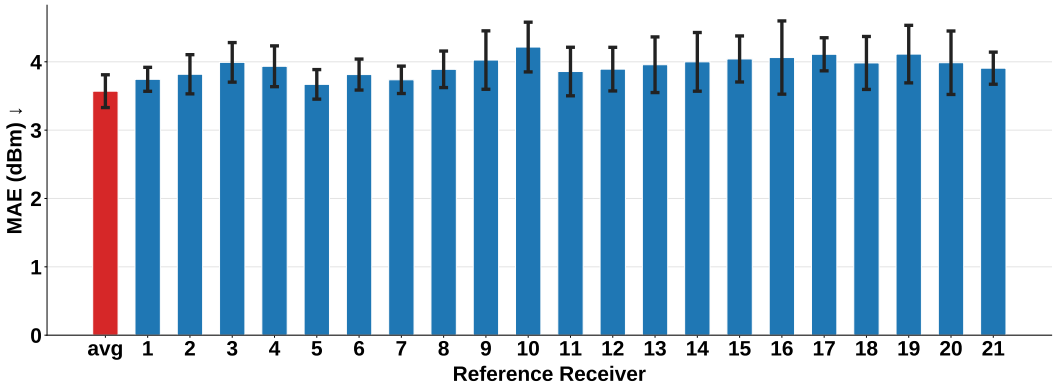


Figure 11: Sensitivity of RxGS to the Stage I reference receiver choice on BLE. Each of the first 21 bars corresponds to using one receiver as the reference, and the `avg` bar uses the averaged signal across all receivers. The single-reference variants span 3.67 to 4.22 dBm, a 0.55 dBm spread, while the averaged-signal initialization achieves 3.57 dBm.

G.7 Application Study: Network Planning

Setting. A common task for a building owner is to choose where to install a small set of access points so that wireless coverage is maximized across all client positions. Without an accurate channel model, the planner is restricted to deploying access points at positions already surveyed. With RxGS, the planner can score unmeasured candidates and select higher-coverage sites.

As in §5.2, we use a 3-fold split of the 21-gateway BLE RSSI dataset into 14 trained gateways and 7 gateways excluded from training. For each fold we randomly sample $K = 5$ surveyed gateways from the 14 trained ones and compare three deployment strategies. SURVEY-ONLY deploys at the 5 surveyed positions with no model and no choice. OURS runs greedy maximum-coverage selection over all 21 candidate positions, using real RSSI for the 5 surveyed gateways and RxGS predictions for the other 16. UPPER BOUND runs the same greedy selection but uses real RSSI at all 21 gateways, giving a full-survey oracle. Once a deployment is chosen, we evaluate it on the test transmitters by computing each transmitter's strongest real RSSI among the 5 deployed access points, which is the standard max-power association in wireless networks. Figure 12 renders one representative fold as a 24×32 grid heatmap with cell size of about 1.6×1.6 m. Each cell is colored by the fraction of transmitters whose strongest-AP RSSI exceeds -80 dBm, and cells with no test transmitters are shown in gray.

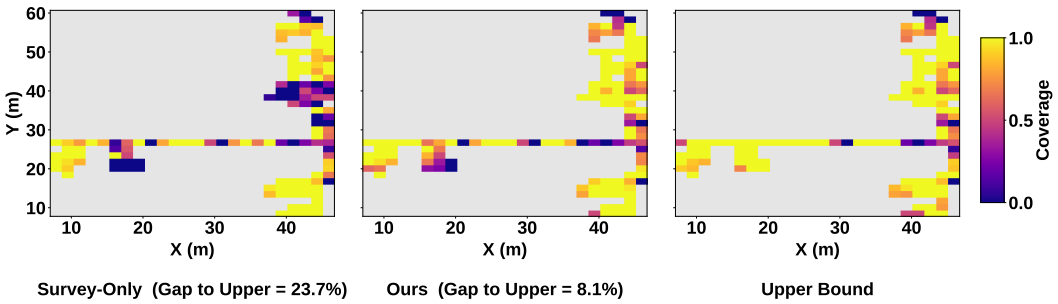


Figure 12: Coverage heatmaps for $K = 5$ access-point deployment on the BLE dataset. Each cell shows the fraction of test transmitters in a $\sim 1.6 \times 1.6$ m region whose strongest-AP RSSI exceeds -80 dBm. Gray cells contain no measurements. SURVEY-ONLY deploys at 5 surveyed gateways, OURS uses RxGS predictions to select 5 from all 21 candidates, and UPPER BOUND greedily selects 5 from full-survey RSSI. For the displayed fold, OURS reduces the gap to UPPER BOUND from 23.7% to 8.1%.

Results. The three panels share the same building footprint and the same evaluation transmitters, yet the resulting coverage patterns differ markedly. SURVEY-ONLY leaves a cluster of low-coverage (dark purple) cells in the upper corridor, where the five randomly surveyed access points provide insufficient coverage. These coverage gaps reflect the restriction to surveyed access-point positions. OURS reduces the low-coverage region by selecting higher-coverage sites among the 16 unmeasured candidates in place of lower-coverage surveyed sites. The remaining low-coverage cells resemble the full-survey reference.

We quantify similarity to the oracle by the average per-cell absolute coverage difference, denoted gap-to-Upper. For the representative fold in Figure 12, the gap falls from 23.7% under SURVEY-ONLY to 8.1% under OURS, a roughly $3\times$ reduction. Aggregating over the 3 folds, OURS raises the share of transmitters covered above -80 dBm from 87.9% under SURVEY-ONLY to 92.6%, recovering about 60% of the gap to the full-survey upper bound of 95.8%. In this study, predicted RSSI expands the candidate set while retaining the same five-gateway survey budget.

Practical Survey Guidance. The measured-to-virtual balance depends on anchor diversity, synthesis fidelity, learnability of receiver effects, and how well the model represents the scene. A practical survey begins with spatially distributed gateways spanning distinct propagation conditions, then uses virtual gateways to densify candidates within the surveyed region. Withholding some measured gateways during model selection can reveal weakly supported areas and guide additional measurements until unseen-gateway accuracy begins to saturate. Figure 10 shows diminishing returns beyond about 12 gateways in this scene. Jointly choosing real and virtual sites remains open.

G.8 Receiver-Level Accuracy Gaps

The RFID spectrum simulation provides visibility labels for examining how the accuracy gap varies with receiver geometry. For each of the 21 receivers, we compare **Ours**$_{\text{GSRF}}$ with the corresponding per-receiver GSRF baseline on the same test samples. We relate the receiver-level PSNR gap to the fraction of transmitters with non-line-of-sight (NLoS) paths, mean grid-probed occlusion, and distance to the nearest training receiver under the protocol of §5.2. For seen receivers, NLoS fraction has the largest observed association with the gap (Pearson $r = 0.43$, $p = 0.054$). Two corner receivers behind furniture, *rx15* and *rx19*, have approximately 99% NLoS pairs and the largest gaps. The correlation is $r = 0.27$ after excluding those two receivers. At unseen receivers, the correlations are $r = 0.23$ for NLoS fraction, $r = 0.12$ for mean probed occlusion, and $r = 0.04$ ($p = 0.88$) for distance to the nearest training receiver. Nearest-anchor distance shows little association with the gap in this split.

G.9 Visibility, Bounce Depth, and Occlusion Features

We use Sionna path geometry to split the same RFID test pairs into line-of-sight (LoS) and non-line-of-sight (NLoS) groups according to last-segment visibility. Table 6 reports PSNR for each

group. For the unseen-receiver comparison, the per-receiver GSRF reference is trained separately with measurements from each evaluated receiver.

Table 6: RFID spectrum PSNR (dB) by last-segment visibility. At unseen receivers, the per-receiver baseline is trained on measurements from each evaluated receiver.

Method	LoS Pairs	NLoS Pairs
Per-receiver GSRF	23.35	23.20
Ours$_{\text{GSRF}}$, seen receivers	22.35	22.59
Ours$_{\text{GSRF}}$, unseen receivers	21.07	21.40

Relative to the per-receiver model on the same pairs, the shared model trails by 1.00 dB on line-of-sight (LoS) and 0.61 dB on NLoS pairs at seen receivers. The corresponding gaps at unseen receivers are 2.28 and 1.80 dB. Among NLoS pairs, even the heaviest blockage-severity tercile has gaps of 0.64 dB seen and 1.50 dB unseen, both below the respective LoS gaps. Grouping pairs instead by dominant-path bounce count gives the largest unseen gap for direct paths at 2.2 dB. The one-, two-, and three-or-more-bounce groups have unseen gaps between 1.6 and 2.0 dB, and the seen groups show no increasing trend with bounce count. In this test set, greater blockage and bounce depth have gaps comparable to or smaller than those of LoS and direct paths, respectively. The PSNR gaps are relative to each group's signal dynamic range.

We also compare the grid-probed last-segment transmittance T_k with Sionna ray-traced visibility. At unseen receivers the probe gives Pearson $r = 0.40$ and an area under the receiver operating characteristic curve (AUC) of 0.70. At seen receivers it gives $r = 0.40$ and AUC 0.73. The probe provides similar, moderate visibility discrimination at seen and unseen receivers.

G.10 Occupancy-Grid Resolution

We retrain Stage II on RFID spectrum at each occupancy-grid resolution, keeping the frozen Stage I geometry fixed. Table 7 shows that all three settings remain within 0.1 dB of the default 128^3 grid. Across the evaluated 64^3 to 256^3 range, PSNR varies by only 0.04 dB, indicating limited sensitivity to grid size for this room.

Table 7: Occupancy-grid resolution on RFID spectrum. Stage II is retrained for each setting.

Grid	64^3	128^3 (default)	256^3
PSNR (dB)	22.43	22.47	22.43

G.11 Conditioning Capacity

We vary the widths of both conditioning MLPs, including the local branch, while keeping their depth at three layers. We retrain Stage II for each RFID setting using the same frozen Gaussian geometry. Table 8 reports nearly constant PSNR across a roughly $25\times$ increase in conditioning parameters. The total PSNR spread is 0.04 dB, and every width remains within 0.1 dB of the default. Stage II takes about 21 minutes per setting, while inference for $N = 21$ receivers remains about 31 ms because the same Gaussian set is rasterized. Widening the three-layer conditioner has little effect on the approximately 0.8 dB seen-receiver spectrum gap to per-receiver GSRF. The CSI configuration already uses width 256, four times the spectrum default, and has a comparable gap. Conditioner depth and alternative designs remain directions for improving this accuracy–efficiency tradeoff.

Table 8: RFID spectrum performance across conditioning widths with three-layer branches retrained in Stage II.

Width d	64 (default)	128	256	512
PSNR (dB)	22.47	22.48	22.44	22.46
Conditioning parameters	11.7k	28.1k	85.6k	298.9k

H Four Evaluation Configurations

RF data synthesis is queried along two axes, transmitter and receiver. Evaluation configurations are distinguished by which axis contains test positions excluded from training. Table 9 enumerates the

four possibilities. The labels in this table follow our experimental setup: transmitter positions are densely sampled, while receivers occupy a smaller set of fixed positions.

Table 9: Four evaluation configurations.

	Seen Transmitter	**Unseen Transmitter**
Seen Receiver	(1) training fit	(2) novel transmitter
Unseen Receiver	(3) novel receiver	(4) novel transmitter & receiver

Prior per-receiver 3DGS methods principally evaluate cell (2), predicting at an unseen position on the moving, densely surveyed side while retaining a checkpoint trained for the fixed receiver side [18, 20, 19]. The endpoint names differ across papers: WRF-GS+ fixes the physical receiver, whereas RF-3DGS and GSRF fix the physical transmitter, which maps to the fixed receiver role in our formulation after applying channel reciprocity. Here, reciprocity aligns endpoint roles for comparing model configurations, while direction-resolved measurements such as angular spectra retain their physical endpoint orientation. In all three designs, the moving side enters rendering, while the fixed side is absorbed into model parameters or training data. NeRF^2 instead queries one shared field from the receiver's ray origin and can support unseen receivers at greater computational cost [5]. GRaF's query uses reference spectra measured at each receiver [33]. We follow this convention in §5.1 for direct comparison, while using one unified model for all N receivers. Table 2 reports the resulting training, inference, and storage savings.

Cell (4) motivates RxGS: it predicts at an unseen transmitter and a fixed receiver excluded from training but located within the surveyed scene and deployment region. This models the addition of a receiver within an existing surveyed environment without collecting its measurements or fitting another receiver-specific checkpoint. We report cell (4) in §5.2. Cell (4) subsumes cell (3), since reusing training transmitters at an unseen receiver is a sub-case of generalizing along both axes. Cell (1) reports training-time fit and is bounded by cell (2).

I Broader Impacts

RxGS reduces the cost of collecting RF data for wireless networking and sensing. Site surveys at thousands of transmitter–receiver positions are labor-intensive and time-consuming. Synthesizing RF data at unmeasured fixed receiver positions inside a static, surveyed environment could reduce the measurement burden for wireless planning and indoor localization, as illustrated in §5.4 and Appendix G.7. The coverage estimates may also guide infrastructure placement.

A future application is connectivity-aware planning for mobile agents within a static, previously surveyed environment. A shared model could score candidate receiver positions inside the surveyed region without fitting a new checkpoint at each location. Future work will evaluate continuous receiver motion, changing environments, and robotic deployments.

The same capability could be misused for passive localization or tracking without consent if an attacker combines synthesized fingerprints with captured signals. This risk is not unique to RxGS and applies to any RF fingerprinting method, but practitioners deploying fingerprint databases should follow standard privacy safeguards such as on-device inference, aggregation, and informed consent.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction reflect the main contributions of RxGS, including receiver-generalizable RF data synthesis within a single model and a radiance-representation-agnostic conditioning design.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: Section 6 discusses the limitations of RxGS, including the static-scene assumption and the lack of cross-scene generalization.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The geometry-radiance decomposition and the last-segment factorization are stated with full assumptions in Section 4 and proved in Appendices A, B, and C.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Appendix E provides full implementation details and hyperparameters, and Appendix F describes the datasets and splits used for all experiments.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The BLE RSSI and WiFi CSI datasets are publicly available as cited, the RFID spectrum dataset generation is fully described in Appendix F, and the code will be released upon publication.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Appendix E specifies the optimizer, learning rates, and all hyperparameters, and Appendix E specifies the data splits for each dataset.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Table 1 and all per-receiver results report the mean and standard deviation across per-receiver means, as described in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section 5 states that all experiments are conducted on a single NVIDIA H100 GPU, and Table 2 reports per-method training and inference times.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms with the NeurIPS Code of Ethics in every respect.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Appendix I discusses the broader impacts of RxGS.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper poses no such risks.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All existing datasets and baselines used in the paper are properly cited in Section 5 and Appendix F.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The RFID spatial spectrum dataset is documented in Appendix F, and the implementation is documented in Appendix E.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development does not involve LLMs.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.