Physics-Unrolled Neural Operator for Accelerating High-Fidelity Wireless Ray Tracing

Rafid Umayer Murshed
University of Illinois Urbana-Champaign
rum3@illinois.edu

Saif Ur Rahman
University of Illinois Urbana-Champaign
saifu2@illinois.edu

Mingyue Tang
University of Illinois Urbana-Champaign
mt55@illinois.edu

Elahe Soltanaghai
University of Illinois Urbana-Champaign
elahe@illinois.edu

Abstract

Site-specific wireless design relies on ray tracing to predict how signal strength varies across a floorplan, but a converged indoor radio map takes tens of seconds to simulate and planning a single building requires thousands of them. A learned surrogate can remove most of that cost, yet training one runs into a second problem: high-fidelity labels are exactly what is too expensive to produce at scale, so the affordable labels come from finite-ray simulations that carry residual Monte Carlo noise. We address this with Ray-tracing Neural Operator (RayNO), a physics-unrolled neural operator that reconstructs the converged output of a ray tracer from one cheap low-fidelity run and scene priors. Rather than treating a radio map as a generic image, RayNO refines the field through three stages matched to reflection, diffraction, and scattering, each supervised by a label of matching physical complexity. We prove that when the label error is conditionally unbiased the learned operator can end up closer to the high-fidelity reference than the labels it was trained on, and we verify that condition directly in the simulator using repeated independent ray seeds. Across diverse floorplans, RayNO outperforms image-to-image baselines, wireless learning models, monolithic neural operators, and classical multi-fidelity regression on both image-quality and wireless deployment metrics, while reducing the cost of a high-fidelity map from roughly $28\,\mathrm{s}$ to $0.11\,\mathrm{s}$.

1 Introduction

Accurate wireless field modeling aims to predict how electromagnetic signals vary across a physical environment. The key challenge is that wireless propagation involves coherent multipath interactions with scene geometry and materials, including reflection, diffraction, and scattering, which can produce sharp spatial variations in received signal strength (RSS) [1, 2]. As a result, practical tasks such as WiFi access-point placement, coverage planning, localization, and network reliability analysis often require high-fidelity radio maps that capture these local variations across space [3, 4, 5]. However, generating such high-fidelity radio maps requires dense ray sampling or detailed propagation modeling, making it computationally expensive[6, 7]. This cost adds up quickly in practice, because design studies such as comparing transmitter locations or floorplan layouts require a new simulation for every candidate configuration.

This motivates the central question of this paper: **can a learned model recover the high-fidelity output of a ray tracer from a low-cost run of the same simulator?** A straightforward direction is to treat this as an image-to-image learning problem [8, 9], where a model maps a low-fidelity input radio map to a sharper output. However, such models are often optimized for global visual similarity

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

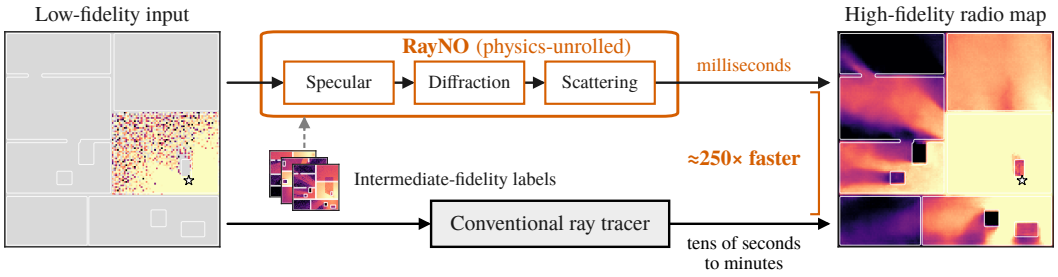


Figure 1: RayNO replaces a conventional ray tracer: three physics-unrolled stages (specular, diffraction, and scattering), trained on intermediate-fidelity labels, turn a low-fidelity input into a high-fidelity radio map roughly $250\times$ faster.

or average pixel-wise accuracy, which can smooth out sharp local variations caused by multipath. This is especially problematic for radio maps because those local variations are not visual artifacts; they reflect underlying propagation effects such as reflection, diffraction, and scattering, and are critical for coverage and planning [1, 2].

Another limitation is that learning wireless fields cannot realistically rely on fully high-fidelity labels, since generating them for large training sets is computationally expensive [7]. Instead, the available labels are typically intermediate-fidelity simulations that contain richer spatial structure than the low-fidelity input, but remain noisy and imperfect, as illustrated in Fig. 1. The learning problem is therefore not simply to fit these labels, but to refine coarse fields using the stable structure present in intermediate-fidelity supervision while avoiding simulation noise.

To address these challenges, we propose RayNO, a physics-unrolled cascade of neural operators for recovering high-fidelity radio maps from low-fidelity ray-tracing outputs. Rather than predicting the full radio map in one step, RayNO refines the output progressively through multiple operator stages, each focusing on a specific wireless propagation effect: broad reflected coverage, edge-driven diffraction near walls and corners, and fine local scattering fluctuations from objects and multipath. Geometry and scene priors are used to constrain these stages, keeping each refinement grounded in the floorplan, material layout, and transmitter/receiver context. This stage-wise design allows the model to build on the output of the previous operator instead of applying one generic smoothing function to the entire field. This operator formulation further enables field-to-field prediction across different layouts and spatial resolutions, while wireless-physics-aware losses encourage the model to preserve meaningful spatial gradients rather than only matching average pixel values.

We evaluate RayNO across diverse indoor scenes using simulated radio maps from NVIDIA SionnaRT [10], comparing it against standard vision architectures, general-purpose neural operators, and wireless-specific baselines. Once trained, RayNO produces a high-fidelity radio map about $250\times$ faster than running the high-fidelity simulation. We show that RayNO can outperform the intermediate-fidelity labels used for training, effectively denoising them without access to high-fidelity reference radio maps during training. Our theoretical analysis identifies a precise condition under which this is possible: the label error must be conditionally unbiased for each scene, meaning that for a fixed floorplan the noise has zero mean and does not systematically bias the field in any direction. The random simulation errors then average out across training samples, while the stable propagation structure shared across scenes remains learnable. This condition holds in our ray-tracing setup, where the intermediate-fidelity labels carry zero-mean Monte Carlo noise. In summary, the paper makes the following contributions:

(i) We introduce RayNO, a three-stage neural-operator cascade for wireless field refinement that learns from noisy intermediate-fidelity supervision and progressively captures spatial structure associated with reflection, diffraction, and scattering in unseen environments.

(ii) We design propagation-aware training objectives that emphasize fine-scale spatial structure in wireless fields, helping preserve details that standard pointwise losses tend to smooth out.

(iii) We introduce a deployment-oriented evaluation protocol based on wireless metrics and show that it reveals coverage and planning differences that standard image-quality metrics miss.

(iv) We prove a denoising theorem that requires no clean reference labels. It shows that, with enough training scenes, an operator trained on noisy labels is provably closer to the high-fidelity reference than those labels themselves. This extends classical image-denoising results from single-image settings to operator learning across many physical scenes.

2 Background and Related work

Wireless Fields and Propagation Mechanisms. A wireless field describes how electromagnetic signal strength varies across a physical environment. The received signal at each location is the superposition of multiple propagation paths, reflecting off of surfaces and objects. A radio map discretizes this field by assigning a received-signal-strength (RSS) value to each location in the deployment area. These maps support decision-making tasks such as access-point placement, coverage planning, localization, and identifying outage-prone regions.

As illustrated in Fig. 2, different propagation mechanisms leave distinct spatial signatures: Line-of-sight (LoS) and specular paths form broad coverage patterns, diffraction creates sharp changes near shadow boundaries, and scattering introduces fine local fluctuations. High-fidelity radio map simulation must therefore capture multiple superimposed mechanisms, not merely increase spatial resolution. Full-wave electromagnetic solvers provide detailed approximations of Maxwell-equation behavior but are often too expensive for repeated room- or building-scale planning [11, 12], which has motivated learned surrogates even for single components such as metasurfaces [13]. In practice, scene-scale wireless design often relies on ray tracing, which is more scalable than full-wave simulation but still trades off cost and fidelity through the ray budget, number of bounces, and modeled propagation mechanisms, with runtime increasing by over $25\times$ in some cases [6, 14, 15].

Many wireless ray-tracing software packages use Monte Carlo (MC) ray tracing, in which propagation is approximated by launching finitely many rays, tracing their interactions with scene geometry, and aggregating their contributions into an RSS field. With fewer rays, the field is cheaper to compute but only partially converged; with more rays, it becomes richer and more stable at higher cost.

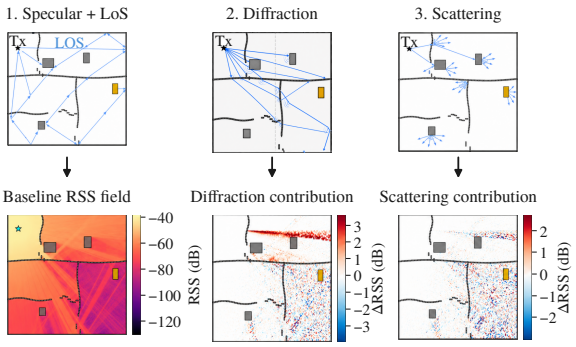


Figure 2: Individual propagation mechanisms jointly shaping the radio map.

Related Work. Our work builds on three strands of scientific machine learning (ML). Neural operators, physics-informed networks, and multi-fidelity methods learn maps between structured fields, often when the supervision is coarse or comes from simulations at different cost [16, 17, 18, 19, 20, 21]; in wireless, operator learning has also been combined with explicit propagation-path structure for channel compression [22]. Image denoising work shows that useful predictions can be learned from noisy labels alone, without any clean reference [23, 24, 25]. A recent wireless paper uses physics priors to learn the statistical distribution of channel parameters from noisy access-point observations [26], and self-supervised training can keep wireless models robust to imperfect channel estimates [27]. We draw on each. None of them targets indoor radio maps, and none separates the three propagation effects — reflection, diffraction, and scattering — that produce the distinct local patterns wireless coverage planning depends on.

Learning-based wireless surrogates predict signal behavior without exhaustive site surveys. RadioUNet and the Time of Arrival (ToA) dataset family train a U-Net that maps scene geometry directly to a pathloss map [8, 9]. More recent work fuses sparse on-site radio frequency (RF) measurements with visual or geometric priors to fit a richer representation of one scene at a time [28, 4, 5]. Both routes assume something we avoid: the first needs clean training labels and learns one fixed image-to-image map; the second needs RF measurements collected on-site at every new scene. We train once across many scenes on noisy ray-traced labels, and predict on unseen floorplans with no on-site RF data.

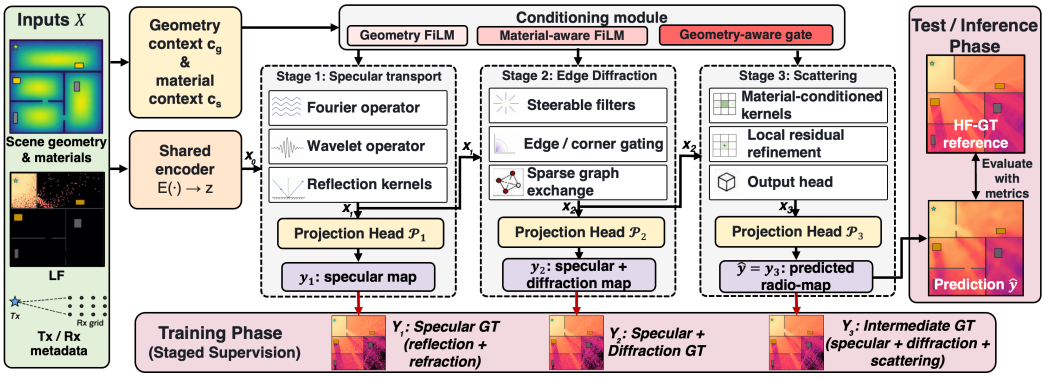


Figure 3: RayNO estimates high-fidelity radio maps using a three-stage neural-operator architecture that captures different wireless propagation mechanisms.

A closely related line uses continuous per-scene representations — NeRF-style fields, neural ray tracers, and radiance-field methods that fit one model to a single scene from on-site measurements and predict signal at any location in that scene [29, 30, 3, 31, 32]. Standard channel models, geometry-based simulators, and differentiable ray tracers provide the physical scaffold these methods sit on [33, 34, 6, 35]. In all of them, ray tracing is either the forward simulator that produces the answer or the target a single neural field is fit against. We use ray tracing the other way around: as the simulator to be accelerated, with its own cheap, noisy outputs serving as supervision across many scenes. Our model learns a prediction that is closer to a high-fidelity reference than those labels are, and it handles reflection, diffraction, and scattering (Fig. 2) in three explicit stages rather than smoothing them into one.

3 Method

3.1 Problem Formulation

Consider a wireless scene with a fixed transmitter (e.g. a WiFi access point) and receivers distributed over a 2D floorplan $\Omega \subset \mathbb{R}^2$. The corresponding radio map is a 2D matrix of received signal strength values, where each cell p represents a receiver location on the floorplan. We denote by $\mathcal{Y}$ the space of such radio maps. For each scene, the model input $X \in \mathcal{X}$ consists of a low-fidelity radio-map (i.e., a coarse simulation-based estimate) and scene priors that describe the floorplan geometry, material information, and coordinate context. The model is trained using a higher-fidelity but still imperfect radio map as the label, denoted by $Y_{\text{IF-GT}} \in \mathcal{Y}$, which we call the *Intermediate-Fidelity radio map labels*. Compared with the low-fidelity input u, this label has higher spatial resolution and captures a broader set of wireless interaction mechanisms (e.g. diffraction and scattering). However, it is still an approximate target rather than the final ground truth. For evaluation, we use a separate held-out high-fidelity reference, denoted by $Y_{\text{HF-GT}} \in \mathcal{Y}$. This *high-fidelity ground-truth radio map reference* represents the desired radio map that the model aims to recover, but it is used only for evaluation and is never seen during training.

The objective is to learn a neural operator $f_\theta : \mathcal{X} \to \mathcal{Y}$ that maps each input scene representation X_i to a predicted radio map $\hat{Y}_i = f_\theta(X_i)$, where θ denotes the trainable parameters. For each training scene $i = 1, \ldots, n$, the available label is the intermediate-fidelity radio map $Y^{(i)}_{\text{IF-GT}}$. The model is optimized as

$$\theta^\star = \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\big(f_\theta(X_i), Y^{(i)}_{\text{IF-GT}}\big), \tag{1}$$

where $\theta^\star$ denotes the learned parameters and $\mathcal{L}$ is the training loss. Although training uses the intermediate-fidelity ground-truth labels, the goal is for $\hat{Y}_i$ to approach the held-out high-fidelity ground-truth reference.

This is achievable under the assumption that the intermediate-fidelity label is a conditionally unbiased approximation of an ideal radio map for each scene. Let Y_{oracle} denote this ideal radio map. Then the

intermediate-fidelity label can be viewed as

$$Y^{(i)}_{\text{IF-GT}} = Y^{(i)}_{\text{oracle}} + \varepsilon_i, \qquad \mathbb{E}[\varepsilon_i \mid X_i] = 0. \tag{2}$$

This means that, for a fixed scene representation X_i, the intermediate-fidelity label may contain noise, but the noise does not consistently overestimate or underestimate the ideal radio map. Under this assumption, training on intermediate-fidelity ground-truth labels does not systematically push the model away from the desired mapping. Instead, the zero-mean errors can average out across training examples, allowing the learned operator to approach the ideal mapping. Since Y_{oracle} is not directly available, we use the held-out high-fidelity reference $Y_{\text{HF-GT}}$ as its evaluation proxy. We formalize this as a denoising theorem that requires no clean reference labels (Appendix H).

3.2 Input Representation

For each scene, we separate the model input into three groups as $X = (u, g, c) \in \mathcal{X}$. where

(i) $u \in \mathcal{Y}$ is the low-fidelity (LF) input radio map, which provides a coarse estimate of the received signal strength. We also include a binary mask that marks which grid cells contain valid low-fidelity estimates, along with local features that describe whether the nearby signal pattern has a clear dominant direction, as expected from ray-like propagation.

(ii) g denotes geometry/material prior maps, which are aligned with the input radio-map grid. These maps encode floorplan and propagation-relevant context, including signed distance fields, obstacle and boundary information, transmitter-distance maps, shadow-region indicators, and material-related maps.

(iii) c denotes coordinate priors, which provide spatial and transmitter/receiver context, including receiver-grid coordinates, transmitter-location cues, and transmitter/receiver height information. These coordinate features are Fourier-expanded to provide explicit spatial encoding.

The three input groups are passed through a shared encoder to form the initial latent field. They are also injected directly into the stage-specific operators, where they help form the propagation-aware residual updates at each stage.

3.3 Three-Stage Physics-Unrolled Operators

Figure 3 illustrates our proposed Physics-Unrolled Neural Operator (RayNO) that refines a low-fidelity radio map into a high-fidelity radio map prediction through a sequence of wireless-aware stages. The key intuition is that wireless propagation effects appear at different levels of spatial complexity: broad coverage patterns are mainly governed by direct and reflected paths, sharper transitions arise near walls, corners, and shadow boundaries due to refraction and diffraction effects, and fine local variations are caused by scattering and multipath interference. Rather than using a single generic operator to recover all of these structures, RayNO builds the prediction progressively through three cascaded residual operators:

$$x_k = x_{k-1} + \tilde{\mathcal{H}}_k(x_{k-1}; g, c, s_\star), \qquad y_k = P_k(x_k) \in \mathcal{Y}, \qquad k = 1, 2, 3, \tag{3}$$

where x_k is the latent field after stage k, $\tilde{\mathcal{H}}k$ is the learned residual operator for that stage, and P_k maps the latent field to the stage-wise radio-map prediction y_k. The stages build on each other through the residual update: stage 1 produces an initial reconstruction of the dominant radio-map structure due to specular reflections, stage 2 refines the remaining edge- and transition-related effects due to diffraction, and stage 3 adds finer local corrections due to scattering. Each intermediate output y_k is supervised by a stage-specific label Y_k with matching physical complexity. In this fidelity ladder, Y_1 and Y_2 provide intermediate supervision, while $Y_3 \equiv Y_{\text{IF-GT}}$ is the final training label used in the problem formulation, and the final prediction is $\hat{Y} = y_3$. The adaptive residual map $s_\star \in [0, 1]^\Omega$, predicted from the first-stage latent field x_1, indicates where the current prediction still contains unresolved structure. This allows the later stages to focus on residual propagation effects that were not captured by the previous operator, rather than refining the entire field uniformly. We next define the details of each stage-specific residual operator.

Stage 1: Specular refinement. The first stage captures the dominant, large-scale structure of the radio map. This structure is mainly determined by open propagation paths, line-of-sight regions,

and strong reflections from major surfaces such as walls. These effects produce smooth, globally coupled coverage patterns. The Fourier basis represents this broad structure compactly, but pure spectral truncation can blur localized features such as wall-cast shadow boundaries. We therefore combine a global spectral operator $\mathcal{G}$ with a learned multiscale path $\mathcal{W}$, similar in structure to a wavelet transform, that preserves localized structure, and a reflection-aware local operator $\mathcal{L}_{\text{refl}}$ whose oriented kernels are aligned with the dominant reflection direction inferred from g:

$$\tilde{\mathcal{H}}_1(x; g, c) = \mathcal{G}(x; g, c) + \mathcal{W}(x) + \mathcal{L}_{\text{refl}}(x; g, c). \tag{4}$$

After this stage, the model estimates $s_\star$ to flag regions needing further refinement. Appendix C.3 gives the layer-level details of all three stages.

Stage 2: Diffraction refinement. The second stage focuses on regions affected by diffraction, such as areas near wall edges, corners, and doorways, where the floor plan causes abrupt propagation changes. These regions are difficult to recover from Stage 1 because diffraction depends on both the local obstacle geometry and the relationship among nearby diffraction sites. We therefore combine a direction-selective diffraction operator $\mathcal{D}_\theta$ with a graph neural operator $\mathcal{N}_K$:

$$\tilde{\mathcal{H}}_2(x; g, c, s_\star) = s_\star \odot \left[\mathcal{D}_\theta(x; g, c) + \lambda \mathcal{N}_K(x; g, c)\right]. \tag{5}$$

Here, $\mathcal{D}_\theta$ applies geometry-guided filters using edge, corner, and shadow-boundary cues from g, while $\mathcal{N}_K$ is a graph neural operator that exchanges information among the top-K candidate diffraction regions. Multiplication by $s_\star$ restricts the update to regions where Stage 1 leaves unresolved structure.

Stage 3: Scattering refinement. The third stage focuses on regions affected by scattering, which occurs when the propagating signal interacts with sharp corners, small geometric irregularities, or materials that disturb the reflected field. Unlike the broad specular structure captured in Stage 1 or the edge-driven diffraction effects refined in Stage 2, scattering produces short-range, fine-scale fluctuations that depend on both local floor-plan geometry and material properties. To capture these effects, we use local kernels $\kappa_s[g]$ generated from the geometry/material input g:

$$\tilde{\mathcal{H}}_3(x; g, c, s_\star) = (\kappa_s[g] * x) \odot (\alpha_0 + \alpha_1 s_\star), \tag{6}$$

where $*$ denotes convolution, $\kappa_s[g]$ is the scattering kernel conditioned on the local geometry/material representation g, and α_0 and α_1 control the baseline and residual-guided strength of the scattering correction.

3.4 Training Loss Functions

The model is trained with a multi-term objective that supervises stage-wise predictions, preserves spatial transitions, and aligns residual updates with propagation-related features:

$$\mathcal{L}_{\text{total}} = \sum_{k=1}^{3} \lambda_k \mathcal{L}_{\text{Huber}}^{(k)} + \lambda_{\text{sob}} \mathcal{L}_{\text{sob}} + \sum_{k=1}^{3} \mu_k \mathcal{L}_{\text{trans}}^{(k)}. \tag{7}$$

Stage-matched Huber $\mathcal{L}_{\text{Huber}}^{(k)}$. Each prediction y_k is compared with its corresponding label Y_k using a Huber loss, which reduces sensitivity to occasional large errors in the intermediate-fidelity labels. A stage-specific confidence map further emphasizes regions where the corresponding propagation effect is expected to be active.

Distance-modulated Sobolev $\mathcal{L}_{\text{sob}}$. To preserve sharp spatial transitions near walls, corners, and shadow boundaries, we penalize gradient mismatch in the final prediction:

$$\mathcal{L}_{\text{sob}} = \mathbb{E}\left[\left\| w_{\text{dist}}(X) \odot (\nabla y_3 - \nabla Y_3) \right\|_2^2\right], \tag{8}$$

where ∇ denotes the spatial gradient and $w_{\text{dist}}(X)$ emphasizes geometrically important regions.

Transport-feature alignment $\mathcal{L}_{\text{trans}}^{(k)}$. Each residual update is compared with the corresponding label update after both are projected onto directional-energy features:

$$\mathcal{L}_{\text{trans}}^{(k)} = \mathbb{E}\left[\left\| \Phi_k(y_k - y_{k-1}) - \Phi_k(Y_k - Y_{k-1}) \right\|_2^2\right], \tag{9}$$

where $\Phi_k(\cdot)$ extracts local descriptors such as amplitude, coherence, and dominant orientation. This encourages residual corrections to follow propagation structure rather than only pixel-wise error.

The model is trained with a short curriculum that introduces stage-wise supervision first, followed by transport-alignment terms and the weights λ_k, λ_{sob}, and μ_k control the relative contributions of different loss factors. Implementation details are provided in Appendix D.

3.5 Connection to the Theory

The theorem in Appendix H requires the model class to approximate the clean radio map well, with approximation error $\varepsilon^2_{\mathrm{approx}}$ small compared with the label noise. The staged design targets this term. Let $f_k^\star$ denote the clean radio map with the propagation effects of stages 1 through k, and let $f_0^\star = y_0 = u$. Each stage then learns one increment $\Delta_k = f_k^\star - f_{k-1}^\star$, and the error of the final prediction is bounded by the errors on these increments:

$$\left\|y_3 - f_3^\star\right\|^2 \le 3 \sum_{k=1}^{3} \left\|\hat{\Delta}_k - \Delta_k\right\|^2, \qquad \hat{\Delta}_k := y_k - y_{k-1}. \tag{10}$$

Each increment has its own spatial structure, and each stage uses an operator suited to it, whereas a monolithic operator has to fit all three with one set of kernels. The stage-wise losses keep the denoising property of a single-target loss (Appendix H.9).

4 Evaluation

4.1 Setup

Dataset. We generate the dataset using the NVIDIA Sionna RT ray-tracing simulator [10] over randomly generated floorplans with varying areas, room counts, layouts, and furniture-like objects. For each scene, the low-fidelity input is generated using 10^4 rays with only single-bounce reflections, providing a coarse and incomplete radio map. Models are trained on intermediate-fidelity labels generated with 10^6 rays and full physics, and evaluated against a held-out high-fidelity reference radio maps, generated with the same full-physics setting but 10^8 rays. The high-fidelity ground truth (HF-GT) reference is never used during training. Test floorplans differ from the training floorplans in room count, layout, and area, allowing us to evaluate generalization to unseen indoor environments.

Baseline models. We compare RayNO against three groups of baselines: image-to-image regressors, including CNN [36], U-Net [37], ResNet [38], and ViT [39]; DL-based wireless channel models, including NERF^2[30], RadioUNet [8], GeNeRT [40], and WiGATr [29]; and monolithic neural operators, including FNO [41], TFNO [42], UNO [43], SFNO [44], CodaNO [45], and WNO [46]. We also include intermediate-fidelity label as a baseline, since it represents the supervisory target used during training and provides a reference for whether a model can improve beyond the noisy labels.

Evaluation metrics. We evaluate predictions using both wireless and image-quality metrics. Wireless metrics focus on whether the predicted map preserves coverage failures, fading behavior, or spectral efficiency. Image-quality metrics, including RMSE, MAE, SSIM, and LPIPS, measure pixel-level accuracy and structural/perceptual similarity. More details on the metrics are provided in Appendix F.

Computational cost. Including the low-fidelity run it takes as input, one RayNO prediction takes about 0.11 s, against about 28 s for a high-fidelity simulation (Appendix D.6).

4.2 Results

Prediction accuracy relative to the high-fidelity reference. Table 1 compares all predictions against the held-out HF-GT reference. Overall, RayNO gives the most accurate reconstruction across both wireless and image-quality metrics. It reduces tail error to 0.10, brings the fading ratio close to the ideal value of 1.0, increases Outage F1 to 0.82, and reduces RMSE to 3.5 dB. These improvements indicate that RayNO is not only reducing average pixel-wise error, but also more accurately captures coverage-critical regions, such as fading and shadowed areas, where wireless performance can degrade. The first row of the table reports the error of intermediate-fidelity radio maps against high-fidelity references. Since intermediate-fidelity radio maps are the training labels,

Table 1: Quantitative radio map prediction results showing that RayNO consistently outperforms all baselines across wireless and computer vision metrics.

		Wireless Performance				Computer Vision / Generic			
Model Family	**Model**	**Outage F1 ↑**	**Fading Ratio ≈ 1**	**Tail Error (5%) ↓**	**MCESE ↓**	**RMSE (dB) ↓**	**MAE (dB) ↓**	**SSIM ↑**	**LPIPS ↓**
SIONNA	Training Labels	0.172	1.782	1.661	1.311	14.012	5.486	0.725	0.476
Computer Vision	CNN	0.490	0.799	1.043	1.443	4.798	3.152	0.904	0.277
	U-Net	0.178	0.815	0.448	1.076	3.851	2.326	0.919	0.266
	ResNet	0.107	0.761	0.531	1.152	4.228	2.692	0.905	0.291
	ViT	0.017	1.495	0.145	1.637	15.310	6.188	0.652	0.475
DL-based Channel Models	NeRF^2	0.026	0.695	1.724	3.405	5.670	3.041	0.857	0.345
	RadioUNet	0.000	0.314	9.356	9.348	10.327	5.960	0.840	0.296
	GeNeRT	0.000	0.608	1.405	2.133	5.620	3.932	0.857	0.373
	WiGATr	0.002	1.517	0.559	2.078	16.305	6.807	0.643	0.487
Neural Operators	FNO	0.025	0.750	0.690	1.605	4.045	2.362	0.903	0.313
	TFNO	0.018	0.715	0.805	1.848	4.125	2.280	0.912	0.280
	UNO	0.030	0.710	0.592	1.671	4.384	2.631	0.898	0.307
	SFNO	0.111	0.719	0.556	1.645	4.467	2.633	0.891	0.320
	CoDA-NO	0.040	0.501	2.080	4.320	7.153	4.776	0.775	0.442
	WNO	0.191	0.809	0.373	0.965	3.982	2.414	0.913	0.272
Proposed Method	**RayNO (Ours)**	**0.820**	**1.013**	**0.100**	**0.747**	**3.471**	**2.086**	**0.928**	**0.229**

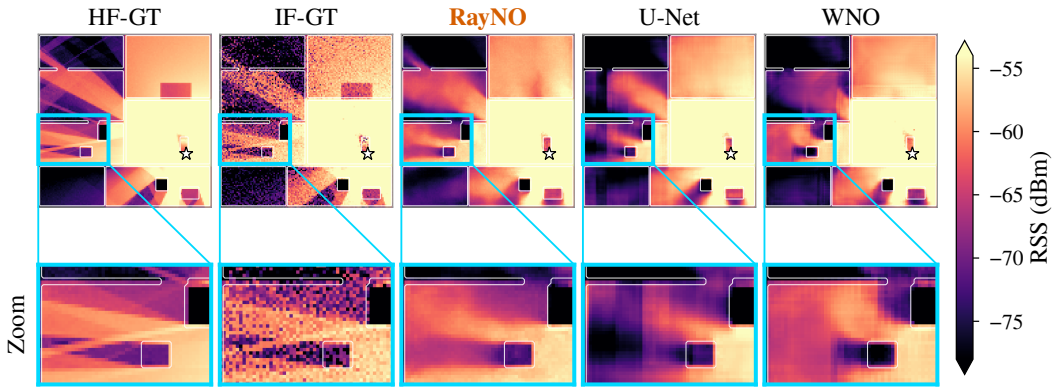


Figure 4: RayNO recovers fine details from noisy labels and approaches the high-fidelity reference. Walls are drawn in white and the star marks the transmitter; the bottom row enlarges the boxed region of each map.

this row serves as a reference for the noise level in the supervision. The fact that RayNO outperforms this row shows that the model learns a prediction closer to the high-fidelity reference than the labels it was trained on. Multi-fidelity co-kriging [47], the per-scene model GSRF [5], and averaging repeated simulations also fall short of RayNO (Appendices E and G.3).

Impact of physics-based model decomposition. Several baselines appear competitive under image-quality metrics. For example, U-Net reaches SSIM 0.919, TFNO reaches 0.912, and WNO is within 15% of RayNO on RMSE. However, their wireless metrics are much weaker: U-Net reaches only 0.18 Outage F1, TFNO drops to 0.018, and RadioUNet predicts almost no outage regions. This shows that visual similarity or low average error does not necessarily mean the model preserves the propagation structures that matter for wireless deployment. Image-to-image models tend to recover the average field appearance, while monolithic neural operators capture some global structure but smooth over regime-specific effects. By decomposing the prediction into specular, diffraction, and scattering refinements, RayNO better preserves these wireless-critical structures. As a downstream example, using radio map prediction for access-point placement on a 10,000 m^2 floorplan gives 92 Mbps throughput for RayNO versus 16 Mbps for CNN, even though the two models differ by only 2.7% in SSIM; details are provided in Appendix G.

Ablation study of model components. Table 2 shows that the main design components contribute to wireless reliability. Removing semantic/material conditioning causes Outage F1 to collapse from 0.82 to 0.001, indicating that geometry alone is insufficient for predicting coverage failures; the model also needs to know how different materials and objects affect propagation. The labels need not be exact: with 10% of them corrupted, RayNO keeps 91% of its Outage F1 (Appendix G.2). The loss design has a similar effect. Removing the distance-aware gradient loss or using only plain Huber supervision reduces Outage F1 to 0.11 and 0.13. The staged architecture is also complementary: specular-only prediction gives Outage F1 0.21, adding diffraction raises it to 0.40, and adding scattering increases it

Table 2: Ablation study showing that RayNO benefits from its staged architecture, geometry priors, and propagation-aware losses.

Category	Variant	Outage F1 ↑	Fading Ratio ≈ 1	Tail Error ↓	RMSE (dB) ↓	SSIM ↑
Learning Paradigm	1A: Single Target (No Curr)	0.637	1.124	0.927	3.787	0.927
	1B: Progressive (No Curr)	0.784	0.956	0.217	3.407	0.926
Loss Formulation	2A: w/o Sobolev Gradient	0.111	0.814	0.381	3.584	0.923
	2B: w/o Transport (Ray)	0.740	0.910	1.273	3.512	0.916
	2C: Unweighted Masked Huber	0.128	0.818	0.197	3.655	0.916
Engineered Priors	3A: w/o Geometric Priors	0.780	0.967	0.110	3.660	0.923
	3B: w/o Semantic Priors	0.001	0.731	0.959	4.107	0.919
	3C: w/o Ray Descriptors	0.791	**0.997**	0.179	3.656	0.927
Architecture Scale	4A: Block 1 Only (Refl)	0.206	0.834	0.197	3.724	0.921
	4B: Blocks 1 & 2 (Refl + Diff)	0.399	0.850	**0.096**	3.603	0.927
Micro-Architecture	4C: w/o Graph Corrector	0.805	0.985	0.098	3.530	0.928
	4D: w/o Local Wavelet Path	0.808	0.992	0.213	3.632	0.927
Proposed Full	**5: Full RayNO (Ours)**	**0.820**	1.013	0.101	**3.471**	**0.928**

to 0.82, with consistent improvements in tail error and fading behavior. Other components, including the curriculum, ray descriptors, and module choices, have smaller but consistent effects.

Qualitative analysis. Figure 4 shows an example radio map comparing the predictions from RayNO and the baselines on a test floorplan. The intermediate-fidelity labels contain some fine propagation signatures, but they are corrupted by Monte Carlo noise and can be sparse in some regions. U-Net and WNO both keep the broad specular backbone but lose narrow streaks, shadow boundaries, and local fluctuations: U-Net introduces blocky artifacts, and WNO blurs the streaks. These are the same coverage-critical details captured by the wireless metrics. In contrast, RayNO recovers these structures with a pattern closer to HF-GT.

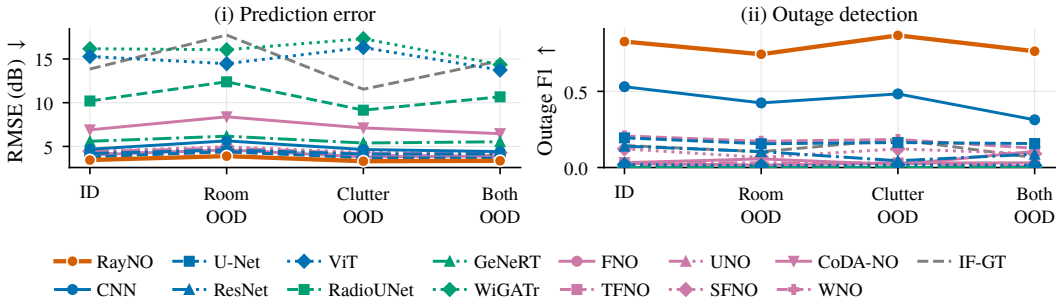
4.3 Sensitivity Analysis

Impact of out-of-distribution (OOD) scene shift. We split the test set based on whether the room count and clutter count fall inside or outside the training range, producing four groups: ID, Room-OOD, Clutter-OOD, and Both-OOD. Figure 5 shows that RayNO achieves the lowest RMSE across all groups, ranging from 3.31 to 3.89 dB, and maintains Outage F1 between 0.74 and 0.87. In contrast, the strongest baseline on each split reaches only 0.34–0.53 Outage F1. This indicates that RayNO generalizes well to unseen layout and clutter, especially in coverage-critical regions.

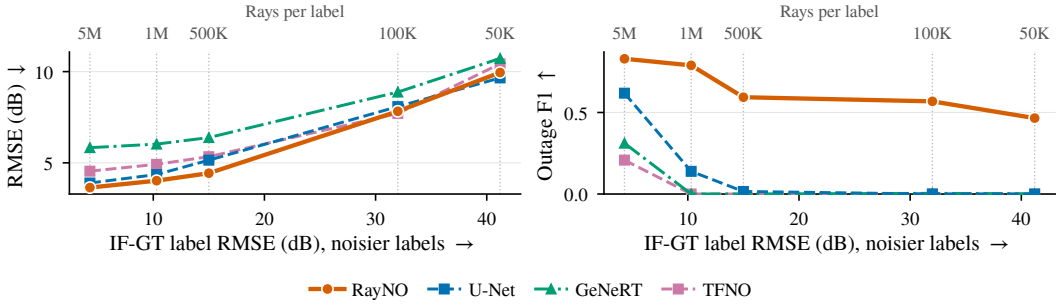
Impact of intermediate-fidelity label ray budget. To evaluate how noisy the training labels can be, we vary the ray budget used to generate intermediate-fidelity radio maps from 50K to 5M rays, while keeping the HF-GT evaluation reference fixed at the 10^8-ray setting. As expected, all models degrade when the training labels become noisier, but Figure 5 shows that RayNO degrades more gracefully. Even at 50K rays, RayNO maintains Outage F1 around 0.47, while all baselines fall below 0.05, and its local-fading ratio remains close to the desired value of 1.0. This result shows that, as long as the intermediate-fidelity labels retain coherent propagation patterns, the physics-aligned stages can extract those patterns while suppressing much of the noise. With 10^6 rays per stage label, RayNO already keeps 95% of the Outage F1 it reaches at 5M rays, so labels of this size suffice in our setting.

Impact of training label noise structure. To evaluate the impact of label-noise type, we replace the original intermediate-fidelity labels with synthetic labels created by adding controlled noise to the HF-GT maps. The models are then evaluated against the uncorrupted HF-GT reference. RayNO is robust to zero-mean noise types, including IID [48], heteroscedastic [49], and spatially correlated noise [50]. Heteroscedastic noise slightly improves RMSE from 3.84 to 3.62 dB and Outage F1 from 0.824 to 0.834, suggesting a regularization effect. In contrast, geometry- and distance-dependent biased noises cause a clear failure, increasing RMSE to 12.06 dB and reducing Outage F1 to 0.359. This shows that the model can suppress unbiased noise, but not systematic bias in the training labels. Averaging up to 512 independent simulations of fixed scenes confirms that the label noise of our simulator is zero-mean (Appendix H.10).

Theoretical Justification. These sensitivity results are supported by our denoising theorem in Appendix H. Under the conditionally unbiased noise model in Equation 2, the expected loss against



(a) In-distribution (ID) and out-of-distribution (OOD) floorplans.



(b) Ray budget of each intermediate-fidelity stage label; the input and the HF-GT reference are fixed.

Figure 5: Model sensitivity to structural distribution shifts and training label noise.

the noisy labels has the same minimizer as the loss against the ideal radio map: zero-mean noise adds variance to the supervision but does not shift the optimal mapping. With an expressive enough operator class and enough training scenes, the model therefore learns the propagation function rather than the label noise.

5 Discussion

Conclusion. This work presents RayNO, a physics-unrolled neural operator for high-fidelity radio map prediction from low-fidelity simulation and scene priors. Instead of treating radio maps as generic images, RayNO decomposes the prediction into propagation-aware stages that progressively refine specular structure, diffraction effects, and scattering-related details. The results show that this design can learn radio map structure that is closer to the held-out high-fidelity reference than the noisy labels used for training, especially on wireless deployment metrics that capture coverage failures, fading behavior, and spectral-efficiency errors. More broadly, the work demonstrates that imperfect simulation labels can still support high-fidelity wireless field prediction when the model architecture and losses are aligned with the underlying physical structures.

Limitations and Future Work. RayNO depends on accurate geometry and material priors; missing floorplan details or unseen materials can introduce systematic errors. It predicts received signal strength only, not phase, channel impulse responses, or MIMO channel matrices. Multiple access points can be handled by running the model once per transmitter, but coherent multi-antenna processing requires phase. Our evaluation covers indoor, single-transmitter scenes with fixed antennas; outdoor scenes, multi-transmitter interference, and antenna patterns remain open. The model also gives point predictions without uncertainty estimates, which matches deterministic ray tracing but would be useful when material properties are uncertain. Finally, all supervision and evaluation are based on ray tracing, so accuracy on a real building is limited by how well the simulator models it. The gap between a simulator and real measurements is mostly systematic, which our noise experiments show the model cannot remove. A practical route is to calibrate the simulator with a few measurements from the target site [35] and train RayNO on the calibrated simulator; validating this on measured data is future work.

Acknowledgments and Disclosure of Funding

We thank the reviewers for their constructive feedback. This work was partially funded by Intel. The authors declare no competing interests.

References

[1] A. Goldsmith, *Wireless communications*. Cambridge university press, 2005.

[2] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. Cambridge university press, 2005.

[3] H. Lu, C. Vattheuer, B. Mirzasoleiman, and O. Abari, "Newrf: A deep learning framework for wireless radiation field reconstruction and channel prediction," in *Proceedings of the 41st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 235, 2024.

[4] X. Chen, Z. Feng, K. Qian, and X. Zhang, "Radio frequency ray tracing with neural object representation for enhanced rf modeling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[5] K. Yang, G. Dong, S. Ji, W. Du, and M. Srivastava, "Gsrf: Complex-valued 3d gaussian splatting for efficient radio-frequency data synthesis," in *Advances in Neural Information Processing Systems*, 2025.

[6] J. Hoydis, F. Aït Aoudia, S. Cammerer, M. Nimier-David, N. Binder, G. Marcus, and A. Keller, "Sionna RT: Differentiable ray tracing for radio propagation modeling," *arXiv preprint arXiv:2303.11103*, 2023. [Online]. Available: https://arxiv.org/abs/2303.11103

[7] C. Modesto, L. Mozart, P. Batista, A. Cavalcante, and A. Klautau, "Accelerating ray tracing-based wireless channels generation for real-time network digital twins," *IEEE Open Journal of the Communications Society*, 2025.

[8] R. Levie, Ç. Yapar, G. Kutyniok, and G. Caire, "Radiounet: Fast radio map estimation with convolutional neural networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 6, pp. 4001–4015, 2021.

[9] Ç. Yapar, R. Levie, G. Kutyniok, and G. Caire, "Dataset of pathloss and ToA radio maps with localization application," *arXiv preprint arXiv:2212.11777*, 2022. [Online]. Available: https://arxiv.org/abs/2212.11777

[10] J. Hoydis, S. Cammerer, F. Ait Aoudia, M. Nimier-David, L. Maggi, G. Marcus, A. Vem, and A. Keller, "Sionna," 2022, https://nvlabs.github.io/sionna/.

[11] Altair, "Altair Feko," https://altair.com/feko, 2026, commercial computational electromagnetics software. Accessed: 2026-05-05.

[12] Ansys, "Ansys HFSS | 3D High Frequency Simulation Software," https://www.ansys.com/products/electronics/ansys-hfss, 2026, commercial full-wave 3D electromagnetic simulation software. Accessed: 2026-05-05.

[13] R. U. Murshed, M. S. A. Rafi, S. Reza, M. Saquib, and I. Mahbub, "MetaFAP: Meta-learning for frequency agnostic prediction of metasurface properties," *arXiv preprint arXiv:2503.14866*, 2025.

[14] Remcom, "Wireless InSite® Propagation Software," https://www.remcom.com/wireless-insite-propagation-software, 2026, commercial wireless propagation and ray-tracing software. Accessed: 2026-05-05.

[15] M. Zhu, L. Cazzella, F. Linsalata, M. Magarini, M. Matteucci, and U. Spagnolini, "Toward real-time digital twins of em environments: Computational benchmark for ray launching software," *IEEE Open Journal of the Communications Society*, vol. 5, pp. 6291–6302, 2024.

[16] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis, "Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators," *Nature Machine Intelligence*, vol. 3, no. 3, pp. 218–229, 2021. [Online]. Available: https://doi.org/10.1038/s42256-021-00302-5

[17] Z. Li, N. B. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. M. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=c8P9NQVtmnO

[18] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019. [Online]. Available: https://doi.org/10.1016/j.jcp.2018.10.045

[19] Z. Li, D. Z. Huang, B. Liu, K. Azizzadenesheli, and A. Anandkumar, "Physics-informed neural operator for learning partial differential equations," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=dtYnHcmQKeM

[20] L. Lu, R. Pestourie, S. G. Johnson, and G. Romano, "Multifidelity deep neural operators for efficient learning of partial differential equations with application to fast inverse design of nanoscale heat transport," *Physical Review Research*, vol. 4, no. 2, p. 023210, 2022. [Online]. Available: https://doi.org/10.1103/PhysRevResearch.4.023210

[21] A. A. Howard, M. Perego, G. E. Karniadakis, and P. Stinis, "Multifidelity deep operator networks for data-driven and physics-informed problems," *Journal of Computational Physics*, vol. 493, p. 112462, 2023. [Online]. Available: https://doi.org/10.1016/j.jcp.2023.112462

[22] R. U. Murshed, S. Hamidi-Rad, E. Soltanaghai, and A. Malhotra, "GCNO: Gramian Chebyshev neural operator for physics-based compression of wireless channels," *arXiv preprint arXiv:2608.18522*, 2026.

[23] J. Lehtinen, J. Munkberg, J. Hasselgren, S. Laine, T. Karras, M. Aittala, and T. Aila, "Noise2noise: Learning image restoration without clean data," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 80, 2018, pp. 2965–2974.

[24] A. Krull, T.-O. Buchholz, and F. Jug, "Noise2void – learning denoising from single noisy images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2129–2137.

[25] Y. Quan, M. Chen, T. Pang, and H. Ji, "Self2self with dropout: Learning self-supervised denoising from single image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1890–1898.

[26] B. Böck, A. Oeldemann, T. Mayer, F. Rossetto, and W. Utschick, "Physics-informed generative modeling of wireless channels," in *Proceedings of the 42nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 267. PMLR, 2025, pp. 4602–4626. [Online]. Available: https://proceedings.mlr.press/v267/bock25a.html

[27] R. U. Murshed, M. S. Ullah, M. Saquib, and M. Z. Win, "Self-supervised contrastive learning for 6G UM-MIMO THz communications: Improving robustness under imperfect CSI," in *IEEE International Conference on Communications Workshops (ICC Workshops)*, Denver, CO, USA, 2024, pp. 220–226.

[28] X. Chen, Z. Feng, K. Sun, K. Qian, and X. Zhang, "RFCanvas: Modeling RF channel by fusing visual priors and few-shot RF measurements," in *Proceedings of the 22nd ACM Conference on Embedded Networked Sensor Systems*, 2024, pp. 464–477. [Online]. Available: https://doi.org/10.1145/3666025.3699351

[29] T. Orekondy, P. Kumar, S. Kadambi, H. Ye, J. Soriaga, and A. Behboodi, "WiNeRT: Towards neural ray tracing for wireless channel modelling and differentiable simulations," in *International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=tPKKXeW33YU

[30] X. Zhao, Z. An, Q. Pan, and L. Yang, "NeRF^2: Neural radio-frequency radiance fields," in *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, 2023. [Online]. Available: https://doi.org/10.1145/3570361.3592527

[31] Y. Bu, J. Yu, K. Zheng, X. Zhang, and P. Pal, "NEAR: Neural electromagnetic array response," in *Proceedings of the 42nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 267. PMLR, 2025, pp. 5749–5774. [Online]. Available: https://proceedings.mlr.press/v267/bu25c.html

[32] S. U. Rahman, R. U. Murshed, A. Dmitriev, C. Tanriover, R. C. Shah, and E. Soltanaghai, "WiNeRF: Measurement constrained radiance fields for actionable wireless channel modeling," in *Proceedings of the International Conference on Embedded Wireless Systems and Networks (EWSN)*, Dresden, Germany, 2026, pp. 1–13.

[33] 3rd Generation Partnership Project (3GPP), "Study on channel model for frequencies from 0.5 to 100 GHz," ETSI, Tech. Rep. TR 38.901, Version 16.1.0, Release 16, 2020. [Online]. Available: https://www.etsi.org/deliver/etsi_tr/138900_138999/138901/16.01.00_60/tr_138901v160100p.pdf

[34] S. Jaeckel, L. Raschkowski, K. Börner, and L. Thiele, "Quadriga: A 3-d multi-cell channel model with time evolution for enabling virtual field trials," *IEEE transactions on antennas and propagation*, vol. 62, no. 6, pp. 3242–3256, 2014.

[35] J. Hoydis, F. Aït Aoudia, S. Cammerer, F. Euchner, M. Nimier-David, S. ten Brink, and A. Keller, "Learning radio environments by differentiable ray tracing," *arXiv preprint arXiv:2311.18558*, 2023. [Online]. Available: https://arxiv.org/abs/2311.18558

[36] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 2002.

[37] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[39] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

[40] K. Bian, M. Tao, S. Sun, and J. Yu, "Genert: A physics-informed approach to intelligent wireless channel modeling via generalizable neural ray tracing," *arXiv preprint arXiv:2506.18295*, 2025.

[41] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," *arXiv preprint arXiv:2010.08895*, 2020.

[42] J. Kossaifi, N. Kovachki, K. Azizzadenesheli, and A. Anandkumar, "Multi-grid tensorized fourier neural operator for high-resolution pdes," *arXiv preprint arXiv:2310.00120*, 2023.

[43] M. A. Rahman, Z. E. Ross, and K. Azizzadenesheli, "U-no: U-shaped neural operators," *arXiv preprint arXiv:2204.11127*, 2022.

[44] B. Bonev, T. Kurth, C. Hundt, J. Pathak, M. Baust, K. Kashinath, and A. Anandkumar, "Spherical fourier neural operators: Learning stable dynamics on the sphere," in *International conference on machine learning*. PMLR, 2023, pp. 2806–2823.

[45] A. Rahman, R. J. George, M. Elleithy, D. Leibovici, Z. Li, B. Bonev, C. White, J. Berner, R. A. Yeh, J. Kossaifi *et al.*, "Pretraining codomain attention neural operators for solving multiphysics pdes," *Advances in Neural Information Processing Systems*, vol. 37, pp. 104 035–104 064, 2024.

[46] T. Tripura and S. Chakraborty, "Wavelet neural operator for solving parametric partial differential equations in computational mechanics problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 404, p. 115783, 2023.

[47] M. C. Kennedy and A. O'Hagan, "Predicting the output from a complex computer code when fast approximations are available," *Biometrika*, vol. 87, no. 1, pp. 1–13, 2000.

[48] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[49] H. White, "A heteroskedasticity-consistent covariance matrix estimator and a direct test for heteroskedasticity," *Econometrica: journal of the Econometric Society*, pp. 817–838, 1980.

[50] N. Cressie, *Statistics for spatial data*. John Wiley & Sons, 2015.

[51] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[52] J. B. Keller, "Geometrical theory of diffraction," 2016.

[53] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[54] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *Journal of machine learning research*, vol. 18, no. 185, pp. 1–52, 2018.

[55] A. Paleyes, M. Pullin, M. Mahsereci, C. McCollum, N. D. Lawrence, and J. Gonzalez, "Emulation of physical processes with Emukit," in *Second Workshop on Machine Learning and the Physical Sciences, NeurIPS*, 2019, arXiv:2110.13293.

[56] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[57] K. Ding, K. Ma, S. Wang, and E. P. Simoncelli, "Image quality assessment: Unifying structure and texture similarity," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 5, pp. 2567–2581, 2020.

[58] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595.

[59] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 2009.

[60] P. L. Bartlett, D. J. Foster, and M. J. Telgarsky, "Spectrally-normalized margin bounds for neural networks," *Advances in neural information processing systems*, vol. 30, 2017.

[61] N. Kovachki, S. Lanthaler, and S. Mishra, "On universal approximation and error bounds for fourier neural operators," *Journal of Machine Learning Research*, vol. 22, no. 290, pp. 1–76, 2021.

[62] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," *Journal of machine learning research*, vol. 3, no. Nov, pp. 463–482, 2002.

A Broader Impact and Limitations

Intended use. RayNO is a learned surrogate for indoor radio-map prediction at sub-6 GHz frequencies, intended for tasks where many candidate configurations must be evaluated against the same scene—access-point placement, coverage planning, and what-if analysis during 6G indoor deployment. The intended user is a network designer or a building-scale digital twin, not an end user of a wireless device. Training and evaluation in this paper use synthetic scenes and synthetic ray-traced fields, so the work raises no privacy, consent, or human-subjects concerns at the dataset level.

Foreseeable benefits. Replacing repeated high-fidelity ray tracing with a learned operator reduces the marginal cost of each candidate configuration by several orders of magnitude; this makes it tractable to consider many more layouts during planning, which we expect to translate into better coverage and lower power in deployed networks. The same surrogate can be used inside outer-loop optimizers for sensor placement or environmental sensing, which would be prohibitive with full ray tracing.

Foreseeable risks. The model is a propagation surrogate; it has no decision-making role and no direct path to harmful downstream uses that ray tracing itself does not already enable. The one risk worth naming is over-trust: a planner who treats the surrogate as ground truth in a regime it was not trained on (e.g., outdoor macrocell, dense metallic clutter, or a frequency outside our training band) may make confident but wrong deployment choices. We address this in two ways: (i) the limitations below state the regime explicitly, and (ii) Section 4.3 shows where the model breaks under controlled perturbation, so that users have a concrete picture of the operating envelope.

Scope and limitations. We list these here, framed by us, so that subsequent appendix sections can be read against a clear scope:

- *Single frequency.* All training and evaluation data are at $5.5\,\mathrm{GHz}$. Frequency transfer would require either retraining or explicit frequency conditioning, which we do not study.
- *Indoor only.* Scenes are bounded $15\,\mathrm{m} \times 15\,\mathrm{m}$ floorplans with concrete outer shell, plasterboard internal walls, and furniture-scale clutter. We do not evaluate outdoor or macrocell propagation.
- *Single transmitter.* Each sample has one active transmitter; we do not study multi-Tx interference or coordinated transmission.
- *Scalar power.* The model predicts received signal strength (RSS), not phase, channel impulse response, angular spectrum, or MIMO matrices. Phase-coherent extension is left to future work.
- *Ray-traced supervision.* Both intermediate-fidelity training labels and HF-GT evaluation references come from the same ray-tracing engine. Sim-to-real calibration against measured radio maps is out of scope.

These are deliberate scoping choices for a single-paper contribution, not fundamental limits of the architecture. Each can be relaxed by changes to the training distribution and conditioning interface; the operator structure and the noise-averaging mechanism in Section H are independent of them.

B Dataset Generation Pipeline

This appendix summarizes the dataset construction details needed to reproduce the experiments. The main text describes the role of the three fidelity levels: the low-fidelity input, the intermediate-fidelity training labels, and the held-out high-fidelity reference. Here we specify how the procedural indoor scenes, ray-tracing fields, and train/test splits are generated.

B.1 Scene generation

Each scene is a $15\,\mathrm{m} \times 15\,\mathrm{m}$ indoor floorplan with a $3\,\mathrm{m}$ ceiling. The outer shell, floor, and ceiling are concrete. Internal room layouts are generated by recursive binary space partitioning (BSP): rectangular rooms are repeatedly split along their longer side, with a minimum room dimension of $3\,\mathrm{m}$ and doorway gaps inserted in internal walls. Figure 6 shows representative training and test floorplans from this generator.

Furniture-scale clutter is added by placing axis-aligned cuboid obstacles inside rooms through rejection sampling. Obstacles have horizontal half-extents sampled from $\mathrm{Unif}(0.3, 0.8)\,\mathrm{m}$ and heights sampled from $\mathrm{Unif}(0.8, 1.5)\,\mathrm{m}$. We enforce a wall margin and a minimum separation between obstacles to avoid degenerate overlaps. Materials are assigned from a small indoor alphabet: concrete for the outer shell, plasterboard for internal walls, and either concrete or metal for furniture. Material parameters follow the corresponding ITU material profiles and are also stored as per-pixel material labels for model conditioning.

B.2 Propagation simulation: LF input, IF-GT labels, and HF-GT reference

All radio maps are generated with Sionna RT [6] at $5.5\,\mathrm{GHz}$ using isotropic transmit antennas. The receive plane is rasterized to a 128×128 grid, and all received-signal-strength (RSS) values are stored in dB. For each transmitter and receive-plane configuration, we generate three fidelity levels:

Low-fidelity input. The low-fidelity field u is generated using 10^4 rays per transmitter with single-bounce specular propagation. This field is used only as an input scaffold and is never used as a training target.

Intermediate-fidelity labels. The training labels are generated using approximately 10^6 rays per transmitter. We store three staged labels with increasing propagation complexity: Y_1 contains the specular subset, Y_2 adds diffraction, and Y_3 adds diffuse scattering. The final intermediate-fidelity label is $Y_{\text{IF-GT}} = Y_3$, while Y_1 and Y_2 provide stage-matched supervision for the RayNO cascade.

High-fidelity reference. The held-out reference is generated with the same propagation mechanisms as Y_3, but with approximately 10^8 rays per transmitter. Thus,

$$\frac{N_{\mathrm{HF}}}{N_{\mathrm{IF}}} \approx 10^2. \tag{11}$$

Under the usual Monte Carlo variance scaling, this makes the HF-GT reference substantially less noisy than the IF-GT training label. HF-GT is used only for validation and testing; it is never used to train RayNO or any baseline.

Cost of generating each fidelity level. Table 3 reports the runtime, memory, and storage profile of the three fidelity levels. The asymmetry is the point of the method: the HF-GT reference dominates the one-time validation and testing cost, while deployment touches only the LF simulation and a 12.1 ms forward pass. Complete staged supervision uses 3×10^6 rays per sample. Simulator usage is 1.96 GB of VRAM per worker; eight parallel workers produced 2,000 complete samples at 44.4 s per sample, and the serialized splits occupy 10.52 GB. The staged labels are generated once and reused, and HF-GT is never a training target. One deployed map therefore costs about 0.112 s against roughly 28 s for a direct HF-GT simulation, a reduction near $250\times$.

Table 3: Cost of generating each fidelity level, per radio map. Runtimes are measured on the generation hardware described above.

Artifact	Rays per map	Role	Runtime per map
LF input u	10^4	Deployment	~ 0.1 s
Each of Y_1, Y_2, Y_3	10^6	Offline supervision	~ 1.8 s
HF-GT reference	10^8	Validation / testing	~ 28 s

B.3 Splits and out-of-distribution subgroups

The training distribution contains scenes with room counts in $\{4, 5, 6, 7\}$ and furniture counts in $\{2, 3, 4, 5, 6\}$. The test distribution widens both ranges to room counts in $\{3, 4, 5, 6, 7, 8\}$ and furniture counts in $\{0, 1, \ldots, 8\}$. This lets us evaluate both in-distribution performance and controlled scene-level distribution shift without changing the underlying floorplan generator.

For the OOD sensitivity analysis in Section 4.3, the test set is partitioned into four disjoint groups:

- *ID*: room and furniture counts both lie inside the training range.
- *Room-OOD*: room count lies outside the training range, while furniture count remains inside.
- *Clutter-OOD*: furniture count lies outside the training range, while room count remains inside.
- *Both-OOD*: both room count and furniture count lie outside the training range.

The released dataset contains the low-fidelity input, staged IF-GT labels, HF-GT references, geometry maps, material labels, transmitter/receiver context, and scene identifiers for each sample.

Dataset Link: RayNO Procedural Indoor RSS Dataset

C Model Architecture

The main text describes RayNO as a three-stage residual operator for specular transport, diffraction, and scattering. This appendix records the architectural details needed to reproduce the reported model, without repeating the full motivation from Section 3.

C.1 Overall structure

RayNO applies three residual operators in sequence. Let x_0 denote the encoded input field and x_k the latent state after stage k. The model uses

$$x_k \;=\; x_{k-1} \;+\; \widetilde{H}_k\big(x_{k-1};\, g, c, s^\star\big), \qquad y_k \;=\; \mathcal{P}_k(x_k), \qquad k = 1, 2, 3, \tag{12}$$

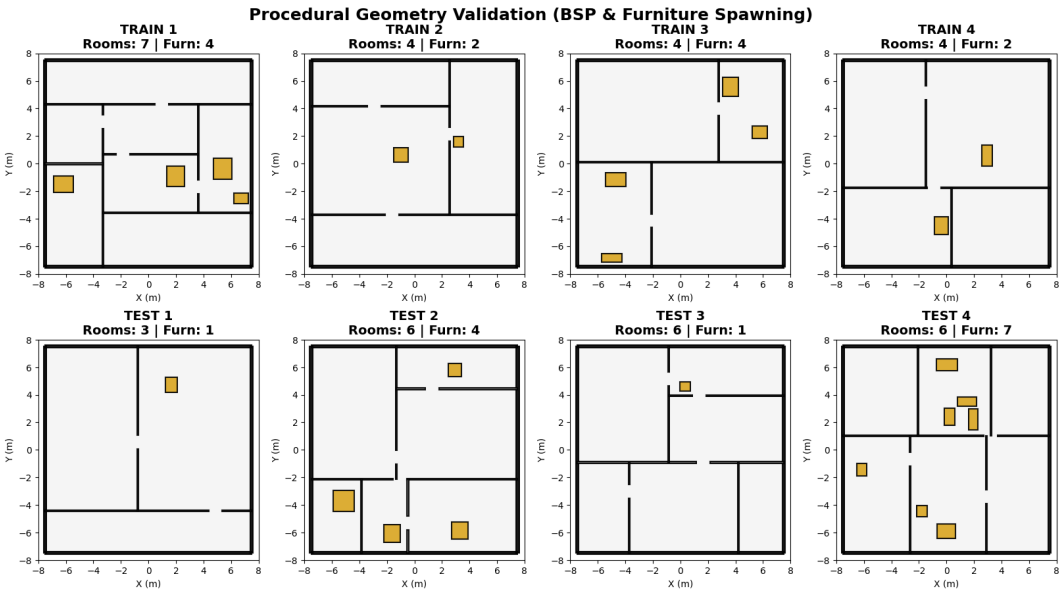


Figure 6: **Representative training and test floorplans.** Environments are procedurally generated via recursive binary space partitioning (BSP) and randomized furniture spawning. The test set introduces structural distribution shifts by expanding the permitted range of room and obstacle counts beyond the training distribution.

where $\widetilde{H}_k$ is the stage-k residual operator, $\mathcal{P}_k$ is the stage readout head, and $s^\star \in [0,1]^\Omega$ is the adaptive sizing field used to restrict later corrections to regions that still need refinement. The final prediction is $\widehat{Y} = y_3$.

The residual form keeps each stage close to an identity correction at initialization and makes the cascade stable: later stages can add localized updates without rewriting the entire field. The model has roughly 2.5M trainable parameters, with most capacity allocated to the first stage because it must recover the global field structure.

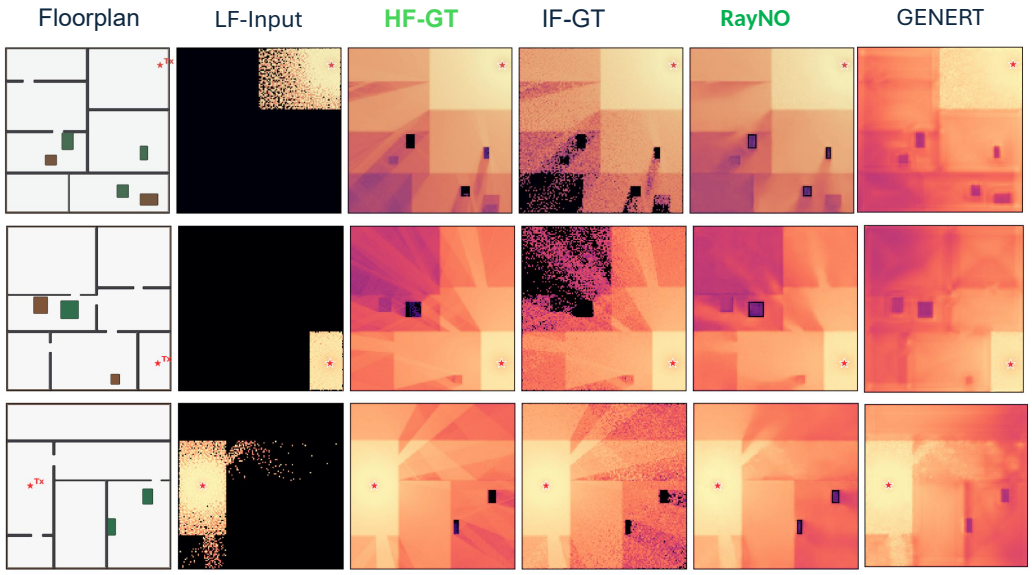
Table 4: Parameter budget by component. Counts are obtained by instantiating the model used for the reported results and summing over the corresponding submodules. Totals are rounded to the nearest thousand.

Component	Trainable parameters
Shared encoder, conditioning, and projection heads	$\sim$ 187K
Stage 1 (specular)	$\sim$ 1.56M
Stage 2 (diffraction)	$\sim$ 614K
Stage 3 (scattering)	$\sim$ 147K
Total	$\sim$ 2.50M

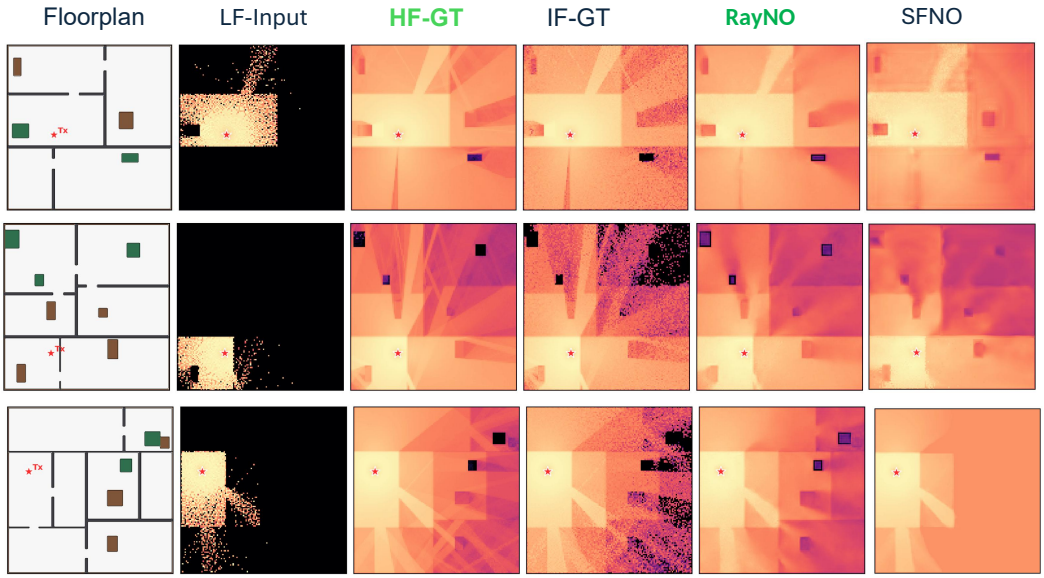
C.2 Encoder and conditioning

The shared encoder maps all input channels to a full-resolution latent field x_0. Its inputs are the low-fidelity RSS field, valid-pixel mask, Fourier-expanded coordinates, Fourier-expanded transmitter-distance features, and a compact ray-evidence descriptor extracted from the low-fidelity input. The encoder is a single 3×3 convolution with normalization and a smooth nonlinearity, producing a latent width of 128.

Two conditioning streams are reused by all stages. The geometry stream combines SDF, transmitter distance, line-of-sight cues, wedge/corner cues, and occupancy. The semantic stream combines free-space/occupied indicators with per-material labels over {free space, plasterboard, concrete,



(a)



(b)

Figure 7: **Qualitative evaluation of radio map reconstruction.** RayNO accurately recovers the High-Fidelity Ground Truth (HF-GT) field from sparse LF-Inputs. Baselines such as GeNeRT (a) and SFNO (b) struggle to preserve these sharp, deployment-critical details and exhibit significant over-smoothing.

metal}. These streams are applied through lightweight modulation blocks rather than by repeated concatenation.

We use three recurring conditioning modules:

- *Geometry FiLM*: a bounded Feature-wise Linear Modulation block conditioned on geometry context [51].
- *Material-aware FiLM*: the same mechanism conditioned on semantic/material context, used where the operator should behave differently across materials.

- *Geometry-aware gate*: a pointwise gate that scales residual updates before they are added back to the latent field.

The modulation magnitudes and gate biases are initialized conservatively so that the network begins close to the low-fidelity input and learns corrections gradually.

C.3 Stage operators

Stage 1: specular refinement. Stage 1 builds the first usable field estimate from the low-fidelity input. It contains three paths: a factorized Fourier path for long-range coupling in the spirit of FNO [17], a local wavelet-style path for localized detail inspired by WNO [46], and a reflection-aware local branch that uses geometry-derived reflection directions to select among oriented filters. The three outputs are fused into a residual update, and the stage readout is added to the low-fidelity field so that y_1 remains anchored to the simulator scaffold.

Stage 1 in detail: what is learned and what is engineered. Each Stage-1 global block applies three paths in parallel. (i) Spectral mixing takes separate one-dimensional FFTs along the horizontal and vertical axes, keeps 20 modes per axis, and applies learned complex weights. (ii) The multiscale path follows an analysis–filter–synthesis chain,

$$\mathrm{PixelUnshuffle}_2 \;\to\; 1{\times}1\,(512{\to}192) \;\to\; 3{\times}3\,\text{depthwise} \;\to\; 1{\times}1\,(192{\to}512) \;\to\; \mathrm{PixelShuffle}_2, \tag{13}$$

which forms a coarser-scale representation, learns localized coarse-to-fine filters there, and reconstructs full-resolution features. We describe this as wavelet-style rather than as a wavelet transform: the analysis and synthesis steps are exact rearrangements and the filters are learned, so the multiscale basis is data-adaptive and no wavelet family is selected. (iii) A pointwise 1×1 branch carries the remainder. Stage 1 additionally uses eight learned 9×9 reflection-aligned filter banks and a learned orientation selector.

Its $1{,}556{,}889$ trainable parameters comprise the spectral weights, the multiscale path, the directional filters, the conditioning and fusion layers, the residual and transport heads, and the gating weights. The quantities that steer these operators are engineered rather than learned: signed-distance normals, reflection directions, boundary cues, directional masks, and ray-evidence kernels are all computed from g before the forward pass.

Stage 2 in detail: the graph operator. The graph corrector $\mathcal{N}_K$ acts on $K = 192$ grid locations selected by the combined adaptive-sizing and geometry scores. Their 128-dimensional grid features are projected to 32-dimensional node states h_i. Edges join spatial neighbours among the selected nodes and carry

$$e_{ij} = \big[\Delta x_{ij},\ \Delta y_{ij},\ d_{ij},\ \Delta w_{ij},\ \Delta c_{ij},\ \Delta b_{ij}^{\mathrm{ISB}},\ \Delta b_{ij}^{\mathrm{RSB}}\big], \tag{14}$$

that is, the relative position and distance between the two nodes together with differences in wall, corner, and incident- and reflection-shadow-boundary cues. Two rounds of message passing then apply

$$m_{ij}^{(t)} = M_\theta\big([h_i^{(t)}, h_j^{(t)}, e_{ij}]\big), \qquad h_j^{(t+1)} = \mathrm{LN}\Big(h_j^{(t)} + U_\theta\big([\,h_j^{(t)},\ \textstyle\sum_{i:(i,j)\in E} m_{ij}^{(t)}\,]\big)\Big). \tag{15}$$

The updated states are projected back to 128 channels, scattered to their grid positions, gated by $s^\star$, and fused with the directional diffraction correction. The branch is zero-initialized, so it begins as the identity and the model recovers the purely directional Stage 2 if training finds no use for it. It holds $21{,}472$ parameters, under 0.9% of the model, which is why removing it costs so little (Table 2, row 4C).

Adaptive sizing field. After Stage 1, a small convolutional head predicts $s^\star \in [0,1]^\Omega$. This field gates the Stage 2 and Stage 3 residual updates. It is not a hand-coded mask; it is learned from the Stage-1 latent, geometry features, and ray-evidence descriptor. Its purpose is to let later stages focus on unresolved regions instead of refining the whole map uniformly.

Stage 2: diffraction refinement. Stage 2 targets edge-driven structure near corners, doorways, and shadow boundaries. It combines a wedge-conditioned directional filtering operator with a sparse graph corrector over geometrically important pixels. The directional operator is aligned with

geometric diffraction intuition [52], while the graph corrector allows nearby hard regions to exchange information. Removing the graph corrector lowers Outage F1 only from 0.820 to 0.805 (Table 2), so a lighter model without it remains a practical option. Both components are gated by $s^\star$ before being added to the latent state.

Stage 3: scattering refinement. Stage 3 models the remaining fine, material-dependent local corrections. It uses depthwise-separable convolutions whose channel-wise scale and shift are conditioned on the per-pixel material embedding:

$$h \;\leftarrow\; \mathrm{GN}(\mathrm{DWConv}(h)) \cdot \big(1+\gamma(m)\big) \;+\; \beta(m), \tag{16}$$

where m is the material embedding map, and $\gamma(m), \beta(m)$ are pointwise functions of that map. This is a local FiLM-conditioned convolution [51]. The update is again gated by $s^\star$ and by the geometry-aware gate.

C.4 Prediction heads and curriculum

The stage readout heads are intentionally small. $\mathcal{P}_1$ and $\mathcal{P}_2$ are 1×1 projections, while $\mathcal{P}_3$ adds a small geometry-conditioned refinement before the final output. Final layers are initialized near zero so that the initial prediction is close to the low-fidelity input.

Training activates the stages progressively. Stage 1 is trained first, Stage 2 is then enabled with its intermediate label Y_2, and Stage 3 is finally enabled with $Y_3 = Y_{\text{IF-GT}}$. The transport-feature auxiliary losses are introduced with a linear ramp only after the main reconstruction objective is established. This curriculum follows the physical ordering of the cascade and is used consistently for all reported RayNO runs.

D Loss Function and Training Protocol

The main text gives the high-level training objective. This appendix specifies the exact loss terms, weights, optimizer, and compute setting used for the reported RayNO results.

D.1 Confidence-modulated reconstruction loss

The intermediate-fidelity labels are finite-budget Monte Carlo ray-tracing outputs, so their reliability varies across space. We therefore use an input-dependent weight that gives extra emphasis to geometrically important regions while masking invalid pixels:

$$w_{\text{conf}}(p) \;=\; \big(1+\mathrm{conf}(p)\cdot \mathrm{geom}(p)\big)\cdot \mathbb{1}_{\text{valid}}(p), \tag{17}$$

with

$$\mathrm{conf}(p) = \frac{1}{1+\big(d(p)/r_{\text{scale}}\big)^2}, \tag{18}$$

$$\mathrm{geom}(p) = \alpha\,\exp\big(-\kappa\,|\,\mathrm{SDF}(p)\,|\big) + \beta\,\mathrm{boundary}(p). \tag{19}$$

Here $d(p)$ is the transmitter distance, $\mathrm{SDF}(p)$ measures distance to geometry, $\mathrm{boundary}(p)$ marks edge/corner/shadow-boundary regions, and $\mathbb{1}_{\text{valid}}$ selects valid IF-GT pixels. The weight depends only on the input and is detached from the optimization graph.

Each stage prediction y_k is supervised by its matching label Y_k using a weighted Huber loss:

$$\mathcal{L}_{\text{rec}}^{(k)} \;=\; \frac{\sum_p w_{\text{conf}}(p)\,\rho_\delta\big(y_k(p)-Y_k(p)\big)}{\sum_p w_{\text{conf}}(p)+\varepsilon}, \tag{20}$$

where ρ_δ is the Huber penalty. The normalization by total active weight keeps the loss scale comparable across scenes.

D.2 Sobolev gradient loss

To discourage over-smoothed predictions, the final prediction is also supervised in the gradient domain. Let D_x and D_y be fixed Sobel operators. We use

$$\mathcal{L}_{\text{sob}} \;=\; \frac{\sum_p w_{\text{sob}}(p)\,\big[\rho_\delta(D_x y_3 - D_x Y_3) + \rho_\delta(D_y y_3 - D_y Y_3)\big]}{\sum_p w_{\text{sob}}(p)+\varepsilon}, \tag{21}$$

with

$$w_{\text{sob}}(p) \;=\; \text{conf}(p) \cdot \mathbb{1}_{\text{valid}}(p). \tag{22}$$

We apply this term only at Stage 3, because the earlier stage targets are intentionally smoother than the final IF-GT field. Appendix H shows that input-measurable weighted Sobolev losses preserve the same population target under conditionally unbiased label noise.

D.3 Transport-feature auxiliary loss

We additionally supervise the residual updates $y_k - y_{k-1}$ using fixed directional descriptors. Let $\Phi_k(\cdot)$ summarize local amplitude, coherence, and dominant orientation from a bank of oriented filters. The transport-feature loss is

$$\mathcal{L}_{\text{trans}}^{(k)} \;=\; \frac{\sum_p w_{\text{ray}}^{(k)}(p)\,\text{conf}(p)\,\rho_{\delta'}\big(\Phi_k(y_k - y_{k-1}) - \Phi_k(Y_k - Y_{k-1})\big)}{\sum_p w_{\text{ray}}^{(k)}(p)\,\text{conf}(p) + \varepsilon}. \tag{23}$$

The descriptor Φ_k is fixed, not learned. Thus this term does not define a second output target; it only encourages each residual update to follow the directional structure of the corresponding label update.

D.4 Composite objective

The full objective is

$$\mathcal{L} \;=\; \sum_{k=1}^{3} \lambda_k\,\mathcal{L}_{\text{rec}}^{(k)} \;+\; \lambda_{\text{sob}}\,\mathcal{L}_{\text{sob}} \;+\; \tau(t) \sum_{k=1}^{3} \mu_k\,\mathcal{L}_{\text{trans}}^{(k)}, \tag{24}$$

where $\tau(t) \in [0, 1]$ is the transport-loss ramp used during curriculum training. The same coefficients are used for every reported RayNO result.

Table 5: Loss hyperparameters used for all reported runs. "Distance scale" refers to r_{scale} in Eq. (18), in physical units relative to the $15\,\text{m} \times 15\,\text{m}$ floorplan.

Term	Symbol	Value
Stage-1 reconstruction	λ_1	0.55
Stage-2 reconstruction	λ_2	1.00
Stage-3 reconstruction	λ_3	1.60
Sobolev gradient	λ_{sob}	0.15
Stage-1 transport feature	μ_1	0.20
Stage-2 transport feature	μ_2	0.28
Stage-3 transport feature	μ_3	0.40
Reconstruction Huber transition	δ	1.0
Transport Huber transition	δ'	0.05
Geometry weighting (wall-proximity coefficient)	α	0.6
Geometry weighting (boundary coefficient)	β	0.4
SDF decay rate	κ	4.0
Distance scale	r_{scale}	$\sim 5\,\text{m}$

D.5 Optimizer, precision, and compute

RayNO is trained with AdamW [53] using $\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 10^{-4}, and base learning rate 8×10^{-4}. The learning rate uses a three-epoch linear warm-up followed by cosine decay. Biases, normalization parameters, and one-dimensional parameters are excluded from weight decay.

Training uses an effective batch size of 64 scenes, mixed precision with `bfloat16`, and gradient clipping at global norm 1.0. A fixed random seed is used for Python, NumPy, and PyTorch. The main results use one seed; robustness to label noise is evaluated separately in Section 4.3.

All reported RayNO runs use a single H200-class NVIDIA GPU. A full run uses 35 epochs over the indoor dataset in Appendix B and takes approximately 1.5 GPU-hours. No multi-GPU or multi-node training is required.

D.6 Inference cost and system-level measurements

Table 6 reports the deployment-side profile. Measurements use `bfloat16` on an H200 NVL with ten warm-up passes followed by fifty synchronized timed runs. The curriculum, the staged labels, and the multi-term objective are training-time constructs only and add no modules at inference; neither the IF-GT labels nor the HF-GT reference is required to predict a map.

Table 6: System-level measurements for RayNO. Compute and latency are per 128×128 radio map.

Quantity	Value
Training	1.5 GPU-hours; 35 epochs; one H200-class GPU
Parameters / model size	2.50M / 12.0 MB
Compute per map	119.24 GFLOPs (59.62 GMACs)
Latency / throughput, batch 1	12.114 ms per map / 82.55 maps s^{-1}
Throughput, batch 32	157.56 maps s^{-1}
Peak inference VRAM, batch 1	294 MiB allocated; 362 MiB reserved
Peak inference VRAM, batch 32	7,925 MiB allocated; 10,836 MiB reserved
End-to-end per map	~ 0.1 s LF simulation + 12.1 ms inference ≈ 0.112 s
Direct HF-GT simulation	~ 28 s

E Baselines and Fair-Comparison Protocol

Table 1 compares RayNO against three baseline families: image-to-image regressors, wireless-specific learning models, and monolithic neural operators. The purpose of this appendix is not to reintroduce these well-known architectures, but to document the comparison protocol and the minimal implementation choices needed for reproducibility.

E.1 Shared protocol

All baselines use the same train/validation/test split, the same IF-GT supervision, and the same HF-GT evaluation reference as RayNO. Each model is sized to the same capacity band as RayNO: the target is 2.5M trainable parameters, and all realized counts lie within $\pm 10\%$ of that target. When a published architecture has a default size outside this range, we adjust only width, depth, mode count, rank, or patch size as appropriate, while preserving the architecture's standard design.

All baselines receive the same information content as RayNO: the low-fidelity field u, geometry/material features g including SDF, transmitter distance, occupancy, and material identity, and coordinate features c including receiver coordinates and transmitter/receiver elevations. These inputs are formatted according to each model family's native interface.

Hyperparameters are selected using the same search protocol for every baseline. We tune learning rate in $\{2 \times 10^{-4}, 6 \times 10^{-4}, 1.2 \times 10^{-3}\}$ and weight decay in $\{10^{-5}, 10^{-4}\}$, giving six candidates per model. We use successive halving [54]: six candidates are trained briefly, the best three are promoted, and then the best two are trained longer. The selected configuration is the one with lowest validation RMSE.

The final training recipe is also shared across baselines: AdamW, batch size 64, mixed precision with `bfloat16`, gradient clipping at global norm 1.0, cosine learning-rate decay, and early stopping on validation RMSE. All reported numbers are produced by one evaluation pipeline that loads each checkpoint, evaluates on the same HF-GT test split, and computes the metrics in Appendix F. Thus, rows in Table 1 differ by architecture, not by data access, optimization, or evaluation code.

E.2 Baseline families

Image-to-image regressors. We include standard dense-prediction architectures: CNN [36], U-Net [37], ResNet [38], and ViT [39]. These models treat the task as generic multi-channel image regression from scene features to an RSS map. Each model predicts a residual correction over the low-fidelity input, which gives a fair starting point because the low-fidelity map already contains useful propagation structure.

Wireless deep-learning baselines. We include wireless-specific models that are closest to the task setting: NeRF^2 [30], RadioUNet [8], GeNeRT [40], and WiGATr [29]. These models are adapted to our 2D indoor RSS-prediction task by replacing their native input/output heads where needed while preserving their core modeling principles. They are trained with the same inputs, parameter budget, and tuning protocol as the other baselines.

Neural-operator baselines. We compare against monolithic neural operators: FNO [41], TFNO [42], UNO [43], SFNO [44], CodaNO [45], and WNO [46]. These are the closest methodological baselines because they also learn field-to-field maps. Unlike RayNO, however, they apply one homogeneous operator family across the whole RSS field rather than separating specular, diffraction, and scattering refinements.

Classical multi-fidelity regression and per-scene fitting. We also compare against two methods outside the three families above. The first is autoregressive co-kriging (AR1), the standard non-neural way to combine a cheap and an expensive simulator [47]. We fit the AR1 multi-fidelity Gaussian process in Emukit [55] on a low-rank coefficient representation of the radio maps. AR1 assumes one global relation between the two fidelity levels, whereas the correction from LF to HF changes across the floorplan with the positions of walls, corners, and materials. The second is GSRF [5], a wireless Gaussian-splatting method with public code and scalar RSSI support, which we adapt to our 128×128 receive plane. GSRF fits a new set of Gaussian primitives for every scene, while every model in Table 1 is trained once and applied to unseen floorplans. We therefore report both methods separately in Table 7. Only the metrics shown were computed for them.

Table 7: Comparison with classical multi-fidelity regression and per-scene fitting, against the held-out HF-GT reference.

Method	RMSE (dB) ↓	SSIM ↑	Fading ratio ≈ 1	Tail error ↓
AR1 MF-GP [47, 55]	13.046	0.646	0.308	1.423
GSRF [5] (fit per scene)	5.070	0.876	0.729	1.157
RayNO (trained once)	**3.471**	**0.928**	**1.013**	**0.101**

E.3 Reference: IF-GT labels

The IF-GT row in Table 1 evaluates the training label itself against HF-GT, with no learned model in between. This row measures the residual error of the supervision signal. It is the reference for the paper's central claim: a learned model that improves over this row is producing a prediction closer to HF-GT than the labels used for training.

E.4 Why rankings differ across metric families

Table 1 shows that strong image-quality scores do not necessarily imply strong wireless performance. RMSE, SSIM, and LPIPS are dominated by the broad, smooth RSS structure, while wireless deployment depends heavily on low-RSS regions, local fading, and cell-edge behavior. This is why several baselines appear competitive under image metrics but fail on Outage F1 or fading ratio. The wireless metrics below are included to expose exactly this failure mode.

F Evaluation Metrics

This appendix defines the evaluation protocol used for Table 1. Standard pixel and perceptual metrics are reported for comparability with dense-regression and image-to-image work. The wireless metrics are defined more explicitly because they measure deployment properties that generic image metrics can miss.

F.1 Reference field and aggregation

All metrics are computed against the held-out HF-GT reference $\widetilde{f}(X)$ from Appendix B, never against the IF-GT training labels. Let $\widehat{Y}$ be a model prediction and $\widetilde{Y} = \widetilde{f}(X)$ be the HF-GT reference, both

in dB on the 128×128 receive-plane grid. Metrics are computed only over valid pixels,

$$\mathcal{M} \;=\; \big\{\, p \in \Omega_{\text{rx}} \;:\; \widetilde{Y}(p) \text{ is finite and } \widetilde{Y}(p) > Y_{\min} + \varepsilon \,\big\}, \tag{25}$$

where $Y_{\min} = -150$ dB is the simulator floor. Pixel metrics are averaged over all valid pixels in the test split. Image-level metrics are computed per scene and then averaged across scenes. The same aggregation is used for every model.

F.2 Pixel and perceptual metrics

We report MAE, RMSE, and PSNR as standard pointwise reconstruction metrics. MAE and RMSE are measured in dB. PSNR uses a fixed dynamic range $R = 170$ dB, corresponding to the interval from the simulator floor -150 dB to the ceiling 20 dB.

F.3 Perceptual metrics

We also report SSIM [56], edge-aware SSIM (ESSIM), DISTS [57], LPIPS [58], and GradMean. SSIM, DISTS, and LPIPS are computed after mapping dB-domain fields to $[0, 1]$; DISTS and LPIPS use three-channel replication and the standard PIQ implementations. ESSIM applies SSIM only near HF-GT edge pixels obtained from Sobel gradients and a Canny detector [59]. GradMean is the mean spatial gradient magnitude of the prediction and is interpreted by closeness to the HF-GT GradMean, not by monotone increase or decrease.

F.4 Wireless deployment metrics

Wireless deployment decisions depend on where coverage holes occur, how much local fading exists, and how accurately the cell-edge tail is predicted. These properties can be hidden by high SSIM or low RMSE, so we report the following wireless-specific metrics. Constants are listed in Table 8.

F.4.1 Outage F1

Outage is defined as RSS below $T_{\text{out}} = -100$ dBm. We threshold both the prediction and HF-GT reference on valid pixels and compute the binary F1 score:

$$\text{F1}_{\text{out}} \;=\; \frac{2 \cdot \text{P} \cdot \text{R}}{\text{P} + \text{R}}, \qquad \text{P} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \qquad \text{R} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{26}$$

where TP, FP, and FN are counted over valid pixels. Outage F1 measures whether the surrogate identifies coverage holes. This is important because a smoothed prediction can have good RMSE while failing to predict weak-signal regions.

F.4.2 Fading ratio (LocalStd9)

For each valid pixel, we compute the local RSS standard deviation in a 9×9 window. Let $\sigma_9(\widehat{Y})(p)$ and $\sigma_9(\widetilde{Y})(p)$ denote this quantity for the prediction and reference. We evaluate fading only in the high-variation subset

$$\mathcal{H} \;=\; \big\{\, p \in \mathcal{M} \;:\; \sigma_9(\widetilde{Y})(p) \geq Q_{0.85}\big(\sigma_9(\widetilde{Y})|_{\mathcal{M}}\big) \,\big\}, \tag{27}$$

and report

$$\rho_{\text{fade}} \;=\; \frac{\text{mean}_{p \in \mathcal{H}}\, \sigma_9(\widehat{Y})(p)}{\text{mean}_{p \in \mathcal{H}}\, \sigma_9(\widetilde{Y})(p)}. \tag{28}$$

The ideal value is 1. Values below 1 indicate oversmoothing, while values above 1 indicate excessive local texture.

F.4.3 SE 5% tail error

We convert RSS to spectral efficiency using a standard AWGN approximation. With bandwidth $B = 20$ MHz, noise figure $F = 7$ dB, and thermal noise -174 dBm/Hz,

$$N_{\text{dBm}} \;=\; -174 + 10 \log_{10}(B) + F. \tag{29}$$

The spectral efficiency at pixel p is

$$\mathrm{SE}(Y)(p) \;=\; \log_2\Big(1 + 10^{(Y(p)-N_{\mathrm{dBm}})/10}\Big) \quad [\mathrm{b/s/Hz}]. \tag{30}$$

On the high-variation mask $\mathcal{H}$, we report the absolute error of the 5th percentile:

$$\big|\mathrm{SE}^{\mathrm{err}}_{5\%}\big| \;=\; \Big|Q_{0.05}\big(\mathrm{SE}(\widehat{Y})|_{\mathcal{H}}\big) - Q_{0.05}\big(\mathrm{SE}(\widetilde{Y})|_{\mathcal{H}}\big)\Big| \quad [\mathrm{b/s/Hz}]. \tag{31}$$

This metric measures whether the model preserves the weakest part of the spectral-efficiency distribution, which is often where deployment decisions are made.

F.4.4 MCESE: mean cell-edge SE error

Within $\mathcal{H}$, we define the cell-edge subset as the bottom 10% of pixels by HF-GT RSS:

$$\mathcal{C} \;=\; \big\{\, p \in \mathcal{H} \;:\; \widetilde{Y}(p) \le Q_{0.10}\big(\widetilde{Y}|_{\mathcal{H}}\big) \,\big\}. \tag{32}$$

MCESE is the mean absolute spectral-efficiency error on this subset:

$$\mathrm{MCESE} \;=\; \frac{1}{|\mathcal{C}|}\sum_{p\in\mathcal{C}} \big|\mathrm{SE}(\widehat{Y})(p) - \mathrm{SE}(\widetilde{Y})(p)\big| \quad [\mathrm{b/s/Hz}]. \tag{33}$$

This directly measures prediction error at the weakest high-variation pixels, where coverage and AP-placement decisions are most sensitive.

F.5 Summary of constants

Table 8: Constants used by the wireless deployment metrics. These values are fixed across all reported experiments and are taken from standard 5–6 GHz indoor wireless practice.

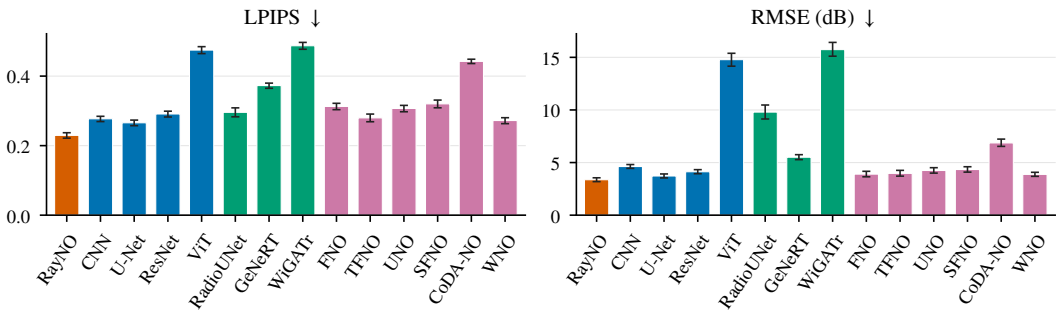
Symbol	Quantity	Value
T_{out}	Outage threshold	-100 dBm
B	Bandwidth	20 MHz
F	Receiver noise figure	7 dB
N_{dBm}	Thermal noise floor (per Eq. (29))	≈ -94 dBm
$Y_{\min}$	Simulator floor for valid mask	-150 dB
R	Dynamic range for PSNR	170 dB
w	Local-std window size	9×9
q_{HV}	High-variance quantile cutoff	0.85
q_{CE}	Cell-edge quantile cutoff	0.10
q_{tail}	SE-tail quantile	0.05

F.6 What the fading ratio does and does not measure

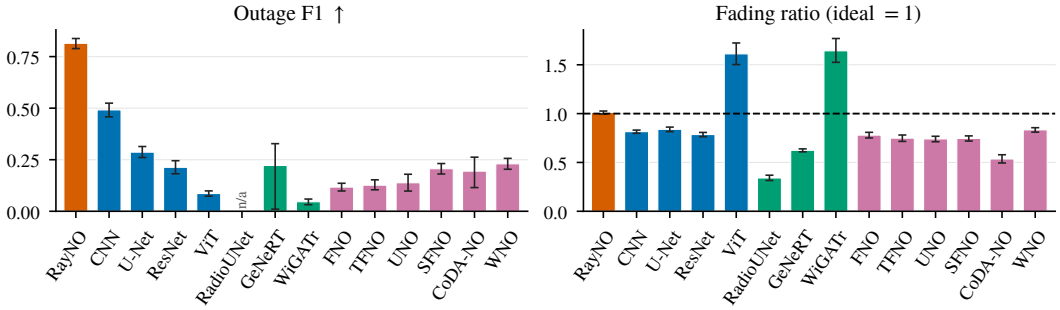
A question that arises when comparing these metrics against a real survey is what an RSS map actually contains. Reflection, diffraction, and scattering are not separate quantities recoverable from each RSS value; they are the mechanisms that shape the map, appearing in a fixed-frequency spatial survey as the broad coverage pattern, the sharp changes near corners and shadow boundaries, and the finer variation between nearby locations. The fading ratio we report measures that spatial variation, not temporal fading from mobility or Doppler, so a survey compared against the model must use comparable time and bandwidth averaging; heavier averaging suppresses precisely the variation the wireless metrics are built to detect.

F.7 Why we report all three families

Pixel, perceptual, and wireless metrics answer different questions. Pixel metrics measure average numerical reconstruction error. Perceptual metrics measure structural similarity when RSS fields are treated as images. Wireless metrics measure whether the prediction supports deployment decisions such as coverage planning, fading-margin estimation, and cell-edge performance assessment. We therefore report all three, but place the most emphasis on the wireless metrics when discussing deployment relevance.



(a) Pixel-level and perceptual metrics (LPIPS, RMSE).



(b) Deployment-critical wireless metrics (Outage F1, fading ratio).

Figure 8: Performance comparison across generic and wireless-specific metrics with 95% bootstrap confidence interval.

Table 9: Extended quantitative comparison on the HF-GT indoor test set. R denotes outage recall. Fad. is the local fading ratio and should be close to 1. SE5 is the absolute 5th-percentile spectral-efficiency tail error in high-variation regions. Grad is the mean spatial RSS-gradient magnitude. The HF-GT reference has GradMean 1.58; hence Grad should be interpreted by closeness to 1.58, not by monotone increase or decrease. Bold marks the best learned predictor.

Family	Model	F1↑	R↑	Fad.≈ 1	SE5↓	MCESE↓	RMSE↓	MAE↓	PSNR↑	SSIM↑	ESSIM↑	LPIPS↓	DISTS↓	Grad≈ 1.58
Label	IF-GT	0.172	0.991	1.782	1.661	1.311	14.012	5.486	21.679	0.725	0.405	0.476	0.294	9.348
CV	CNN	0.490	0.348	0.799	1.043	1.443	4.798	3.152	30.989	0.904	0.548	0.277	0.202	1.380
CV	U-Net	0.178	0.100	0.815	0.448	1.076	3.851	2.326	32.897	0.919	0.551	0.266	0.199	1.388
CV	ResNet	0.107	0.058	0.761	0.531	1.152	4.228	2.692	32.085	0.905	0.509	0.291	0.211	1.368
CV	ViT	0.017	0.045	1.495	0.145	1.637	15.310	6.188	20.909	0.652	0.157	0.475	0.336	10.313
Wireless	NeRF^2	0.026	0.015	0.696	1.725	3.406	5.671	3.041	27.291	0.857	0.399	0.345	0.203	1.935
Wireless	RadioUNet	0.000	0.000	0.314	9.356	9.348	10.327	5.960	24.329	0.840	0.325	0.296	0.268	0.432
Wireless	GeNeRT	0.000	0.000	0.608	1.405	2.133	5.620	3.932	29.615	0.857	0.401	0.373	0.247	1.530
Wireless	WiGATr	0.002	0.005	1.517	0.559	2.078	16.305	6.807	20.362	0.643	0.132	0.487	0.347	10.282
NO	FNO	0.025	0.013	0.750	0.690	1.605	4.045	2.362	32.472	0.903	0.443	0.313	0.225	1.406
NO	TFNO	0.018	0.009	0.715	0.805	1.848	4.125	2.280	32.300	0.912	0.439	0.280	0.217	1.138
NO	UNO	0.030	0.015	0.710	0.592	1.671	4.384	2.631	31.771	0.898	0.415	0.307	0.228	1.256
NO	SFNO	0.111	0.060	0.719	0.556	1.645	4.467	2.633	31.608	0.891	0.384	0.320	0.234	1.228
NO	CoDA-NO	0.040	0.029	0.501	2.080	4.320	7.153	4.776	27.519	0.775	0.204	0.442	0.283	1.848
NO	WNO	0.191	0.108	0.809	0.373	0.965	3.982	2.414	32.606	0.913	0.552	0.272	0.198	1.413
Ours	RayNO	**0.820**	**0.852**	**1.013**	**0.100**	**0.747**	**3.471**	**2.086**	**33.800**	**0.928**	**0.576**	**0.229**	**0.179**	**1.582**

G Extended Results

Table 9 gives the full metric version of the main comparison against the held-out HF-GT reference. It extends Table 1 by adding outage recall, PSNR, ESSIM, DISTS, and GradMean. GradMean is included as a global roughness diagnostic: the HF-GT reference has GradMean 1.58, so this column should be interpreted by closeness to 1.58, not by monotone increase or decrease.

The additional columns reinforce the main-text conclusion without changing it: RayNO is the best learned predictor across the reported metrics, and its GradMean of 1.582 closely matches the HF-GT

Table 10: Engineering interpretation of outage detection for AP placement on a $10{,}000\ \mathrm{m}^2$ enterprise floorplan. The assumed true hard-coverage area is $1{,}000\ \mathrm{m}^2$.

Model	Outage F1 ↑	Recall ↑	Missed area (m^2)	Extra APs	Extra cost
CNN	0.490	0.348	652	3	\$4,500
GeNeRT	0.000	0.000	1,000	4	\$6,000
WNO	0.191	0.108	892	4	\$6,000
RayNO	**0.820**	**0.852**	**148**	**1**	**\$1,500**

Table 11: Engineering interpretation of 5% tail spectral-efficiency error as achieved cell-edge throughput. The target edge throughput is 100 Mbps and the assumed bandwidth is 80 MHz.

Model	SE5 error ↓	Throughput loss	Achieved edge throughput ↑
CNN	1.043	83.4 Mbps	16.6 Mbps
GeNeRT	1.405	112.4 Mbps	0.0 Mbps
WNO	0.373	29.8 Mbps	70.2 Mbps
RayNO	**0.100**	**8.0** Mbps	**92.0** Mbps

value of 1.58. By contrast, IF-GT has GradMean 9.348, indicating that the training labels contain much more high-frequency Monte Carlo roughness than the HF-GT reference. This supports the central claim that RayNO does not simply copy the IF-GT label texture.

Figure 7 provides qualitative examples of the same behavior, and Figure 8 reports bootstrap uncertainty for the main metric families.

G.1 Downstream Case Study: Access-Point Placement

This case study translates the wireless metrics into a simple AP-placement interpretation. It is not a separate benchmark; it is an illustrative calculation using the metrics already reported in Table 9. We consider a $10{,}000\ \mathrm{m}^2$ enterprise indoor floorplan at 5.5 GHz and compare RayNO with one representative baseline from each family: CNN, GeNeRT, and WNO.

Coverage holes and extra APs. Assume that 10% of the floorplan lies in hard-coverage regions that must be detected before final AP placement. Thus,

$$A_{\mathrm{out}} = 0.10 \times 10{,}000 = 1{,}000\ \mathrm{m}^2.$$

Using outage recall, the missed area is

$$A_{\mathrm{miss}} = (1 - \mathrm{Recall}) A_{\mathrm{out}}.$$

Assume one additional enterprise AP can correct approximately $250\ \mathrm{m}^2$ of missed hard-coverage area and that the installed retrofit cost is \$1,500 per AP:

$$N_{\mathrm{retrofit}} = \lceil A_{\mathrm{miss}}/250 \rceil .$$

Under these assumptions, RayNO leaves $148\ \mathrm{m}^2$ of missed hard-coverage area, corresponding to one retrofit AP. CNN leaves $652\ \mathrm{m}^2$, while GeNeRT and WNO leave $1{,}000\ \mathrm{m}^2$ and $892\ \mathrm{m}^2$, respectively.

Achieved cell-edge throughput. For an 80 MHz enterprise WiFi channel, an error of 1 bit/s/Hz corresponds to an 80 Mbps throughput loss. With a 100 Mbps cell-edge target, we use

$$R_{\mathrm{achieved}} = \max\left(0,\ 100 - 80 \times \mathrm{SE5}\right)\ \mathrm{Mbps}.$$

RayNO achieves an estimated cell-edge throughput of 92.0 Mbps, compared with 16.6 Mbps for CNN, 0.0 Mbps for GeNeRT, and 70.2 Mbps for WNO.

Fading and latency risk. The fading ratio indicates whether the model preserves local multipath fluctuation strength. RayNO has fading ratio 1.013, close to the ideal value of 1. CNN, GeNeRT, and WNO have fading ratios 0.799, 0.608, and 0.809, respectively. Thus, these baselines under-estimate fading by roughly 19–39%, while RayNO over-estimates it by only 1.3%. For AP planning, this means RayNO is less likely to make locally unstable regions appear artificially safe.

G.2 Robustness to material-label error

Ablation 3B in Table 2 removes the material map entirely and shows the information is necessary. It does not show how much labeling error the model tolerates, which is the question that matters when the floorplan comes from a building survey rather than from the generator. We therefore took the full trained checkpoint, left it unchanged, and corrupted a fraction of the per-pixel material labels at inference only, reassigning each corrupted pixel to a different class from the indoor alphabet.

Table 12: Material-label robustness. The full trained checkpoint is evaluated unchanged while a fraction of the per-pixel material labels is corrupted at inference. "F1 kept" is Outage F1 relative to the uncorrupted run.

Corrupted labels	Outage F1 ↑	F1 kept	RMSE (dB) ↓	SSIM ↑
0%	0.820	100.0%	3.471	0.928
10%	0.746	90.9%	3.459	0.927
20%	0.609	74.3%	3.592	0.925
50%	0.062	7.5%	4.376	0.911

The overall map is stable throughout: RMSE moves by less than 0.15 dB up to 20% corruption and SSIM by 0.003. Outage F1 decays faster because it is a thresholded quantity, so a small local error near a coverage boundary moves pixels across the outage decision while barely changing the average error. Performance breaks down only once roughly half the material map is wrong, at which point the input no longer describes the same building. Because the model was never retrained under corruption, these numbers are a conservative estimate of what retraining with label noise would achieve.

G.3 Averaging intermediate-fidelity runs: the cost–quality curve

The simplest non-neural way to reduce Monte Carlo noise is to run the intermediate-fidelity simulation several times with different seeds and average the results. Averaging only helps if the noise is zero-mean, so this experiment also checks the noise model. Every improvement along this curve, however, requires running the simulator again.

We generated independent 10^6-ray Stage-3 maps for a fixed scene by changing only the Sionna seed, averaged them in linear power, and measured quality against the HF-GT reference. Table 13 reports the trend.

Table 13: Quality and cost of averaging N independent IF realizations of the same scene, against the held-out HF-GT reference. Runtime is simulator time; the final column is relative to the 28 s HF-GT simulation. RayNO is shown for comparison and needs one LF simulation plus a 12.1 ms forward pass.

Method	RMSE (dB) ↓	Outage F1 ↑	Runtime	Runtime / HF-GT
IF mean, $N = 1$	7.910 ± 0.112	0.371 ± 0.009	1.8 s	$0.064\times$
IF mean, $N = 2$	6.177 ± 0.084	0.471 ± 0.010	3.6 s	$0.129\times$
IF mean, $N = 5$	4.903 ± 0.072	0.583 ± 0.012	9.0 s	$0.321\times$
IF mean, $N = 10$	4.308 ± 0.066	0.637 ± 0.013	18.0 s	$0.643\times$
IF mean, $N = 20$	3.890 ± 0.056	0.667 ± 0.013	36.0 s	$1.286\times$
RayNO	$\mathbf{3.471}$	$\mathbf{0.820}$	$\mathbf{0.112}$ s	$\mathbf{0.004}\times$

The improvement is monotonic, which is what an unbiased estimator should do and is consistent with the repeated-seed evidence in Appendix H.10. The cost, however, grows linearly. By $N = 20$ the averaging has spent 36 s, more than the 28 s a single HF-GT simulation would have taken, and it is still behind RayNO on both RMSE (3.890 against 3.471 dB) and Outage F1 (0.667 against 0.820). Averaging buys accuracy by re-paying the simulator cost that the surrogate is designed to remove.

H Theoretical appendix: proofs for denoising without clean references

This appendix provides the formal statement, proof, and supporting results for the denoising-without-clean-references claim stated informally in Section 3, Eq. (2). After the roadmap (Section H.1), we

introduce setup and notation (Section H.2), state the assumptions (Section H.3), justify conditional unbiasedness of finite-ray Monte Carlo ray tracing (Section H.4), prove the population decomposition (Section H.5), and prove Theorem 4 (Section H.6). We then include two extensions: a finite-dimensional realizable analogue (Section H.7) and a Sobolev target-preservation result supporting the training objective in Section 3.4 (Section H.8).

H.1 Roadmap

Theorem 4 formalizes when empirical risk minimization on noisy IF-GT labels can recover the clean field and eventually become closer to the high-fidelity reference than the labels themselves. Proposition 1 connects the noise model to finite-budget ray tracing, and Lemma 3 gives the key risk-decomposition identity. Proposition 6 provides a simple realizable analogue, while Proposition 7 and Corollary 8 extend the same argument to the discrete Sobolev and weighted Sobolev losses used in Section 3.4.

H.2 Setup and notation

We consider an indoor wireless scene with one transmitter and many possible receiver locations on a two-dimensional receive plane $\Omega \subset \mathbb{R}^2$. The quantity of interest is the received signal strength (RSS) field,

$$Y : \Omega \to \mathbb{R},$$

where $Y(p)$ is the received power, in dB, at receiver location $p \in \Omega$. We predict this scalar power field only; we do not model phase, channel impulse responses, angular spectra, or MIMO channel matrices.

The physical difficulty is that the RSS field is not determined only by distance from the transmitter. Walls, corners, doorways, furniture, and materials create multiple propagation paths. Some paths travel directly or reflect from large surfaces, producing broad coverage trends. Other paths bend around corners or shadow boundaries, producing sharp transitions. Small objects and material changes add local fluctuations. RayNO is built around this simple decomposition: global transport, edge-driven correction, and local scattering correction.

A single input instance is denoted

$$X = (u, g, c) \in \mathcal{X}.$$

Here u is a cheap low-fidelity RSS field from ray tracing, used as a physical scaffold rather than as the training label. The term g denotes geometry and material information aligned with the receive plane, such as wall and obstacle maps, distance-to-transmitter maps, shadow or line-of-sight cues, and material labels. The term c denotes coordinate information, including receiver-grid coordinates and transmitter/receiver height information. The output space $\mathcal{Y}$ is the space of RSS fields on Ω equipped with the norm used in the analysis below.

Let

$$f^\star : \mathcal{X} \to \mathcal{Y}$$

denote the clean field operator: for an input scene X, $f^\star(X)$ is the ideal RSS field that would be obtained by averaging over the simulator's ray-sampling randomness. This clean field is the object we would like to recover, but it is not observed directly.

Training uses a finite-budget ray-tracing label

$$Y = f^\star(X) + \varepsilon,$$

where ε is the simulation noise caused by using a finite number of sampled rays. Evaluation uses a much higher-budget reference

$$\tilde{f}(X) = f^\star(X) + \delta_{\mathrm{abs}},$$

where δ_{abs} is the remaining error of the high-fidelity reference relative to the clean simulator limit. The key asymmetry is that the training label is noisier than the evaluation reference. Intuitively, the IF-GT label is a noisy photograph of the field, while the high-fidelity reference is a much cleaner photograph of the same underlying object.

We learn an operator $f_\theta : \mathcal{X} \to \mathcal{Y}$ from independent training examples $(X_i, Y_i)_{i=1}^n$ by empirical risk minimization:

$$\hat{f}_n \in \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell_f(X_i, Y_i), \qquad \ell_f(X, Y) := \|f(X) - Y\|_{\mathcal{Y}}^2.$$

Theorem 4 below explains when this procedure can produce a predictor closer to the high-fidelity reference $\tilde{f}(X)$ than the training label Y itself. The informal reason is simple: if the simulation noise is conditionally zero-mean, then fitting many noisy labels can recover their average field rather than their sample-specific noise.

Notation summary. Table 14 consolidates the symbols used throughout the appendix. We follow the convention that hatted quantities ($\hat{f}$, $\hat{Y}$) denote model predictions, tilded quantities ($\tilde{f}$) denote High-fidelity references, and starred quantities ($f^\star$) denote the unobservable clean field. Two expectation symbols appear in the proofs: $\mathbb{E}[\,\cdot\,]$ denotes expectation over a fresh test point (X, Y), while $\mathbb{E}_S[\,\cdot\,]$ denotes expectation over the i.i.d. training sample $S = (X_i, Y_i)_{i=1}^n$.

Table 14: Comprehensive notation. Symbols are used consistently across this appendix.

Symbol	Meaning
Domains and spaces	
$\Omega \subset \mathbb{R}^2$	2D floorplan/deployment domain
$\mathcal{Y}$	Separable Hilbert space of RSS fields on Ω
$\langle \cdot, \cdot \rangle_{\mathcal{Y}}$, $\| \cdot \|_{\mathcal{Y}}$	Inner product and norm on $\mathcal{Y}$
$\mathcal{G}$	Space of scene encodings (geometry, materials) on Ω
$\mathcal{C}$	Space of coordinate priors (Rx grid, Tx/Rx elevations)
$\mathcal{X} = \mathcal{Y} \times \mathcal{G} \times \mathcal{C}$	Input space
$X = (u, g, c) \in \mathcal{X}$	Input instance, with law P_X
$u \in \mathcal{Y}$	LF input (10^4-ray, single-bounce specular)
$g \in \mathcal{G}$	Scene encoding (walls, materials, obstacles)
$c \in \mathcal{C}$	Coordinate priors (Rx grid; Tx/Rx elevations)
Fields and labels	
$f^\star : \mathcal{X} \to \mathcal{Y}$	Clean field operator (population limit of MC RT)
$Y \in \mathcal{Y}$	IF-GT label (10^6-ray, full physics)
$\tilde{f}(X) \in \mathcal{Y}$	HF-GT reference (10^8-ray, full physics)
$\varepsilon = Y - f^\star(X)$	MC noise; $\mathbb{E}[\varepsilon \mid X] = 0$
$\delta_{\mathrm{abs}}(X) = \tilde{f}(X) - f^\star(X)$	Reference residual
$\sigma_{\mathrm{MC}}^2 = \mathbb{E}\|\varepsilon\|_{\mathcal{Y}}^2$	MC noise floor
$\eta_{\mathrm{abs}}^2 = \mathbb{E}\|\delta_{\mathrm{abs}}(X)\|_{\mathcal{Y}}^2$	Reference variance
Learning	
$\mathcal{F}$	Hypothesis class (a set of operators $\mathcal{X} \to \mathcal{Y}$)
$f_\theta \in \mathcal{F}$	Parameterized neural operator
$\hat{f}_n$	ERM solution from n i.i.d. samples
$\ell_f(x, y) = \|f(x) - y\|_{\mathcal{Y}}^2$	Squared loss
$\mathcal{L}(f) = \mathbb{E}\,\ell_f(X, Y)$	Population risk
$\widehat{\mathcal{L}}_n(f) = \frac{1}{n}\sum_{i=1}^n \ell_f(X_i, Y_i)$	Empirical risk
$\mathbb{E}_S[\,\cdot\,]$	Expectation over training sample $S = (X_i, Y_i)_{i=1}^n$
$\varepsilon_{\mathrm{approx}}^2 = \inf_{f \in \mathcal{F}} \mathbb{E}\|f - f^\star\|_{L^2(P_X;\mathcal{Y})}^2$	Approximation error
$\mathcal{L}_{\mathcal{F}} = \{\ell_f : f \in \mathcal{F}\}$	Loss class
$\mathfrak{R}_n(\mathcal{L}_{\mathcal{F}})$	Rademacher complexity of $\mathcal{L}_{\mathcal{F}}$
$C_{\mathcal{F}}, M$	Complexity and boundedness constants
Architecture (Section 3)	
$\tilde{\mathcal{H}}_k$ $(k = 1, 2, 3)$	Stage operator (specular / diffraction / scattering)
$x_k, y_k = P_k(x_k)$	Stage-k latent state and field readout
Y_k $(k = 1, 2, 3)$	Stage label; $Y_3 = Y$
E	Shared encoder; $x_0 = E(u, c_g, c_s)$
$c_g \in \mathcal{C}_g, c_s \in \mathcal{C}_s$	Geometry / semantic context (derived from g)
$s^\star \in [0, 1]^\Omega$	Adaptive sizing field
$p \in \Omega$	Spatial coordinate

For any measurable $h : \mathcal{X} \to \mathcal{Y}$, we write $\|h\|_{L^2(P_X;\mathcal{Y})}^2 := \mathbb{E}_X\|h(X)\|_{\mathcal{Y}}^2$.

Bridge to main-paper notation. For accessibility, the main paper writes the ideal field as $Y^{(i)}_{\text{oracle}}$, the IF-GT training label as $Y^{(i)}_{\text{IF-GT}}$, and the held-out high-fidelity reference as $Y^{(i)}_{\text{HF-GT}}$. The translation to the operator-learning notation used in this appendix is:

$$f^\star(X_i) \;\equiv\; Y^{(i)}_{\text{oracle}}, \qquad Y_i \;\equiv\; Y^{(i)}_{\text{IF-GT}}, \qquad \tilde{f}(X_i) \;\equiv\; Y^{(i)}_{\text{HF-GT}}.$$

The conditional unbiasedness assumption stated in Eq. (3) of the main paper, $\mathbb{E}[\varepsilon_i \mid X_i] = 0$, is exactly Assumption (A1) below. The training objective in Eq. (1) of the main paper, with $\mathcal{L}$ taken to be squared loss, is the empirical risk $\widehat{\mathcal{L}}_n(f)$ defined here. We use the operator-learning notation throughout the appendix because it makes the noise-averaging mechanism algebraically transparent.

H.3 Standing assumptions

We begin with the following set of assumptions:

(A1) *Noise model.* $Y = f^\star(X) + \varepsilon$ with $\mathbb{E}[\varepsilon \mid X] = 0$ and $\sigma^2_{\text{MC}} := \mathbb{E}\|\varepsilon\|^2_{\mathcal{Y}} < \infty$.

(A2) *Reference model.* $\tilde{f}(X) = f^\star(X) + \delta_{\text{abs}}(X)$ with $\eta^2_{\text{abs}} := \mathbb{E}\|\delta_{\text{abs}}(X)\|^2_{\mathcal{Y}} < \infty$.

(A3) *Bounded loss class.* There exists $M < \infty$ such that $0 \le \ell_f(x, y) \le M$ for all $f \in \mathcal{F}$ and almost every (x, y).

(A4) *Polynomial Rademacher complexity.* $\mathfrak{R}_n(\mathcal{L}_\mathcal{F}) \le C_\mathcal{F}/\sqrt{n}$ for some $C_\mathcal{F} < \infty$.

Assumption (A1) is justified physically and mathematically in Section H.4 below. Assumption (A3) is a standard truncation-of-loss assumption; in practice RSS values lie in a bounded range (we clip to $[-150, 20]$ dB), which combined with bounded-weight neural operators ensures the loss is uniformly bounded. Assumption (A4) holds for any neural operator class of bounded depth, width, and weight norm with Lipschitz activations by standard norm-based covering arguments [60]; specifically for FNO-style architectures with bounded spectral truncation, $C_\mathcal{F}$ is polynomial in the architectural constants [61].

H.4 MC RT as a Hilbert-valued unbiased estimator

We first verify that the noise model (A1) is mathematically consistent with finite-budget MC RT as used to generate IF-GT labels.

Proposition 1 (Hilbert-valued unbiased MC estimator)**.** *Fix $X = x \in \mathcal{X}$. Let (Ξ, ν_x) denote the path space of propagation trajectories at scene x under the simulator's physics setting, and let $G_x : \Xi \to \mathcal{Y}$ map a path ξ to its field contribution. Assume that $f^\star$ is the Bochner integral*

$$f^\star(x) = \int_\Xi G_x(\xi)\, d\nu_x(\xi). \tag{34}$$

Let q_x be a sampling distribution with $\nu_x \ll q_x$ and define $Z_x(\xi) := \frac{d\nu_x}{dq_x}(\xi)\, G_x(\xi)$. Suppose $\mathbb{E}_{q_x}\|Z_x(\xi)\|^2_{\mathcal{Y}} < \infty$. Given i.i.d. samples $\xi_1, \dots, \xi_m \sim q_x$, the estimator $Y_m(x) := \frac{1}{m}\sum_{r=1}^m Z_x(\xi_r)$ satisfies

$$\mathbb{E}[Y_m(x) \mid X = x] = f^\star(x), \quad \mathbb{E}[\|Y_m(x) - f^\star(x)\|^2_{\mathcal{Y}} \mid X = x] = \tfrac{1}{m}\,\mathbb{E}_{q_x}\|Z_x(\xi) - f^\star(x)\|^2_{\mathcal{Y}}. \tag{35}$$

Proof. By Radon–Nikodym and Bochner integrability, $\mathbb{E}[Z_x(\xi) \mid X = x] = \int_\Xi \frac{d\nu_x}{dq_x}(\xi)\, G_x(\xi)\, dq_x(\xi) = \int_\Xi G_x(\xi)\, d\nu_x(\xi) = f^\star(x)$, giving the first claim. For the second, expand $\|Y_m(x) - f^\star(x)\|^2_{\mathcal{Y}}$ as a double sum and use that distinct samples ξ_r, ξ_s are conditionally independent and centered, so cross-terms vanish:

$$\mathbb{E}[\|Y_m(x) - f^\star(x)\|^2_{\mathcal{Y}} \mid X = x] = \tfrac{1}{m^2}\sum_{r=1}^m \mathbb{E}[\|Z_x(\xi_r) - f^\star(x)\|^2_{\mathcal{Y}} \mid X = x] = \tfrac{1}{m}\,\mathbb{E}_{q_x}\|Z_x(\xi) - f^\star(x)\|^2_{\mathcal{Y}}. \tag{36}$$

□

Remark 2. Setting $\varepsilon := Y_m(X) - f^\star(X)$ recovers Assumption (A1) with $\sigma^2_{\text{MC}} \le \frac{1}{m}\sup_x \mathbb{E}_{q_x}\|Z_x - f^\star(x)\|^2_{\mathcal{Y}}$. The variance therefore decays as $\mathcal{O}(m^{-1})$ in the ray budget. Choosing $m = 10^6$ for IF-GT and $m = 10^8$ for HF-GT gives $\eta^2_{\text{abs}}/\sigma^2_{\text{MC}} \approx 10^{-2}$, consistent with the regime $\eta^2_{\text{abs}} \ll \sigma^2_{\text{MC}}$ used in the main text.

H.5 Population decomposition

The next lemma is the basic identity behind Theorem 4: under unbiased MC supervision, the noisy population risk equals the clean-field risk plus an additive constant.

Lemma 3 (Population decomposition)**.** *Under (A1), for every measurable $f : \mathcal{X} \to \mathcal{Y}$,*

$$\mathcal{L}(f) = \|f - f^\star\|^2_{L^2(P_X;\mathcal{Y})} + \sigma^2_{\mathrm{MC}}. \tag{37}$$

Hence the population minimizer over all measurable f is exactly $f^\star$, and the minimizer over $\mathcal{F}$ is the $L^2(P_X;\mathcal{Y})$-projection of $f^\star$ onto $\mathcal{F}$.

Proof. Decompose $f(X) - Y = (f(X) - f^\star(X)) - \varepsilon$ and expand the squared norm:

$$\|f(X) - Y\|^2_{\mathcal{Y}} = \|f(X) - f^\star(X)\|^2_{\mathcal{Y}} - 2\langle f(X) - f^\star(X),\, \varepsilon\rangle_{\mathcal{Y}} + \|\varepsilon\|^2_{\mathcal{Y}}. \tag{38}$$

The cross-term vanishes in expectation by the tower property and (A1): $\mathbb{E}\langle f(X) - f^\star(X), \varepsilon\rangle_{\mathcal{Y}} = \mathbb{E}\langle f(X) - f^\star(X), \mathbb{E}[\varepsilon \mid X]\rangle_{\mathcal{Y}} = 0$. Taking expectations of the remaining two terms yields (37). □

H.6 Denoising without clean references

We now state and prove the main result.

Theorem 4 (Denoising without clean references)**.** *Assume (A1)–(A4). Let $\hat{f}_n$ be the empirical risk minimizer over $\mathcal{F}$ from n i.i.d. training samples (X_i, Y_i), and let $\varepsilon^2_{\mathrm{approx}} := \inf_{f\in\mathcal{F}} \|f - f^\star\|^2_{L^2(P_X;\mathcal{Y})}$ denote the approximation error of $\mathcal{F}$. Then:*

(i) Clean-field bound.

$$\mathbb{E}_S\|\hat{f}_n - f^\star\|^2_{L^2(P_X;\mathcal{Y})} \le \varepsilon^2_{\mathrm{approx}} + \frac{4C_{\mathcal{F}}}{\sqrt{n}}. \tag{39}$$

(ii) HF-GT bound.

$$\mathbb{E}_S\|\hat{f}_n - \tilde{f}\|^2_{L^2(P_X;\mathcal{Y})} \le 2\varepsilon^2_{\mathrm{approx}} + \frac{8C_{\mathcal{F}}}{\sqrt{n}} + 2\eta^2_{\mathrm{abs}}. \tag{40}$$

(iii) Crossover. *If $2\varepsilon^2_{\mathrm{approx}} + \eta^2_{\mathrm{abs}} < \sigma^2_{\mathrm{MC}}$, then for every $n \ge n^\star := (8C_{\mathcal{F}}/\Delta)^2$ with $\Delta := \sigma^2_{\mathrm{MC}} - 2\varepsilon^2_{\mathrm{approx}} - \eta^2_{\mathrm{abs}}$,*

$$\mathbb{E}_S\|\hat{f}_n - \tilde{f}\|^2_{L^2(P_X;\mathcal{Y})} \;\le\; \mathbb{E}\|Y - \tilde{f}\|^2_{\mathcal{Y}}. \tag{41}$$

In words: the learned operator converges to the clean field at the standard $\mathcal{O}(n^{-1/2})$ rate (39), and beyond the finite sample size $n^\star$ its predictions are at least as close to the high-fidelity reference as the noisy training labels themselves are (41). The crossover condition $2\varepsilon^2_{\mathrm{approx}} + \eta^2_{\mathrm{abs}} < \sigma^2_{\mathrm{MC}}$ formalizes the asymmetric-label-quality regime: the model class must be expressive enough relative to the high-fidelity reference quality ($2\varepsilon^2_{\mathrm{approx}} + \eta^2_{\mathrm{abs}}$ small), and the training noise must be large enough relative to that reference (σ^2_{MC} large). Both sides of this inequality are controllable: the left through architecture and reference budget, the right through the IF-GT ray budget.

Proof of Theorem 4. Fix any $\varepsilon' > 0$ and choose $f_{\varepsilon'} \in \mathcal{F}$ with $\mathcal{L}(f_{\varepsilon'}) \le \inf_{f\in\mathcal{F}} \mathcal{L}(f) + \varepsilon'$. By Lemma 3 applied to $f_{\varepsilon'}$ and to the infimum,

$$\inf_{f\in\mathcal{F}} \mathcal{L}(f) = \inf_{f\in\mathcal{F}} \|f - f^\star\|^2_{L^2(P_X;\mathcal{Y})} + \sigma^2_{\mathrm{MC}} = \varepsilon^2_{\mathrm{approx}} + \sigma^2_{\mathrm{MC}}. \tag{42}$$

Step 1: Oracle inequality. Under (A3)–(A4), the standard Rademacher symmetrization bound [62] gives

$$\mathbb{E}_S\Big[\sup_{f\in\mathcal{F}} \big|\mathcal{L}(f) - \widehat{\mathcal{L}}_n(f)\big|\Big] \le 2\,\mathfrak{R}_n(\mathcal{L}_{\mathcal{F}}) \le \frac{2C_{\mathcal{F}}}{\sqrt{n}}. \tag{43}$$

Since $\hat{f}_n$ minimizes $\widehat{\mathcal{L}}_n$ over $\mathcal{F}$,

$$\begin{aligned}\mathbb{E}_S[\mathcal{L}(\hat{f}_n)] &\le \mathbb{E}_S[\widehat{\mathcal{L}}_n(\hat{f}_n)] + \frac{2C_{\mathcal{F}}}{\sqrt{n}} \le \mathbb{E}_S[\widehat{\mathcal{L}}_n(f_{\varepsilon'})] + \frac{2C_{\mathcal{F}}}{\sqrt{n}} \\ &\le \mathcal{L}(f_{\varepsilon'}) + \frac{4C_{\mathcal{F}}}{\sqrt{n}} \le \inf_{f\in\mathcal{F}} \mathcal{L}(f) + \varepsilon' + \frac{4C_{\mathcal{F}}}{\sqrt{n}}. \end{aligned} \tag{44}$$

Step 2: Clean-field bound. Apply Lemma 3 to $\hat{f}_n$ and subtract σ^2_{MC}:

$$\mathbb{E}_S\|\hat{f}_n - f^\star\|^2_{L^2(P_X;\mathcal{Y})} \le \varepsilon^2_{\mathrm{approx}} + \varepsilon' + \frac{4C_{\mathcal{F}}}{\sqrt{n}}. \tag{45}$$

Since $\varepsilon' > 0$ was arbitrary, taking $\varepsilon' \to 0$ yields (39).

Step 3: HF-GT bound. Decompose $\hat{f}_n(X) - \tilde{f}(X) = (\hat{f}_n(X) - f^\star(X)) - \delta_{\mathrm{abs}}(X)$ and apply $\|a-b\|^2 \le 2\|a\|^2 + 2\|b\|^2$ pointwise. Taking expectations and using (39) gives (40).

Step 4: Label identity. Write $Y - \tilde{f}(X) = \varepsilon - \delta_{\mathrm{abs}}(X)$ and expand. The cross-term vanishes by the tower property: $\mathbb{E}\langle \varepsilon, \delta_{\mathrm{abs}}(X)\rangle_{\mathcal{Y}} = \mathbb{E}\langle \mathbb{E}[\varepsilon \mid X], \delta_{\mathrm{abs}}(X)\rangle_{\mathcal{Y}} = 0$. Therefore

$$\mathbb{E}\|Y - \tilde{f}(X)\|^2_{\mathcal{Y}} = \sigma^2_{\mathrm{MC}} + \eta^2_{\mathrm{abs}}. \tag{46}$$

Step 5: Crossover. If $2\varepsilon^2_{\mathrm{approx}} + \eta^2_{\mathrm{abs}} < \sigma^2_{\mathrm{MC}}$, set $\Delta := \sigma^2_{\mathrm{MC}} - 2\varepsilon^2_{\mathrm{approx}} - \eta^2_{\mathrm{abs}} > 0$ and $n^\star := (8C_{\mathcal{F}}/\Delta)^2$. For $n \ge n^\star$, $8C_{\mathcal{F}}/\sqrt{n} \le \Delta$, so (40) gives $\mathbb{E}_S\|\hat{f}_n - \tilde{f}\|^2_{L^2(P_X;\mathcal{Y})} \le \sigma^2_{\mathrm{MC}} + \eta^2_{\mathrm{abs}} = \mathbb{E}\|Y - \tilde{f}\|^2_{\mathcal{Y}}$, which is the regime (41). □

Remark 5 (On the $\eta^2_{\mathrm{abs}} \ll \sigma^2_{\mathrm{MC}}$ regime). Combining Remark 2 with (46) gives $\mathbb{E}\|Y - \tilde{f}\|^2_{\mathcal{Y}} \approx \sigma^2_{\mathrm{MC}}$ to leading order, justifying the main-text statement that HF-GT acts as an effectively noiseless reference relative to IF-GT.

H.7 Realizable feature-regression analogue

The following proposition makes the noise-averaging mechanism transparent in a finite-dimensional realizable model. It is not used in the proof of Theorem 4; we include it because it produces the same $1/\sqrt{n}$ scaling with explicit constants and clarifies what the abstract Rademacher bound is doing.

Proposition 6 (Exact feature regression)**.** *Suppose $f^\star(x) = \beta^{\star\top}\phi(x)$ for whitened features $\phi(x) \in \mathbb{R}^m$ with $\mathbb{E}[\phi(X)\phi(X)^\top] = I_m$, and that labels obey $Y_i = \beta^{\star\top}\phi(X_i) + \varepsilon_i$ with $\mathbb{E}[\varepsilon_i \mid X_i] = 0$ and $\mathbb{E}[\varepsilon_i^2 \mid X_i] \le \sigma^2$. Let $\hat{\Sigma}_n := \frac{1}{n}\sum_i \phi(X_i)\phi(X_i)^\top$ and assume $\lambda_{\min}(\hat{\Sigma}_n) \ge \mu > 0$. Then the OLS estimator $\hat{\beta}$ satisfies*

$$\mathbb{E}[\|\hat{\beta} - \beta^\star\|_2^2 \mid X_{1:n}] \le \frac{m\sigma^2}{\mu n}, \qquad \mathbb{E}[\|\hat{f} - f^\star\|^2_{L^2(P_X)} \mid X_{1:n}] \le \frac{m\sigma^2}{\mu n}, \tag{47}$$

where $\hat{f}(x) := \hat{\beta}^\top\phi(x)$.

Proof. The normal equations give $\hat{\beta} - \beta^\star = \hat{\Sigma}_n^{-1}(\frac{1}{n}\sum_i \phi_i\varepsilon_i)$. Conditioning on $X_{1:n}$, sample independence and conditional zero-mean make cross-terms vanish, so

$$\begin{aligned}\mathbb{E}[\|\hat{\beta} - \beta^\star\|_2^2 \mid X_{1:n}] &= \tfrac{1}{n^2}\sum_i \mathbb{E}[\varepsilon_i^2 \mid X_i]\,\phi_i^\top\hat{\Sigma}_n^{-2}\phi_i \le \tfrac{\sigma^2}{n^2}\operatorname{tr}\big(\hat{\Sigma}_n^{-2}\textstyle\sum_i \phi_i\phi_i^\top\big) \\ &= \tfrac{\sigma^2}{n}\operatorname{tr}(\hat{\Sigma}_n^{-1}) \le \tfrac{m\sigma^2}{\mu n}. \end{aligned} \tag{48}$$

Whitening gives $\|\hat{f} - f^\star\|^2_{L^2(P_X)} = \|\hat{\beta} - \beta^\star\|_2^2$. □

This shows that even in the simplest setting, with m effective parameters and per-sample variance σ^2, the estimation error decays as $m/(\mu n)$ — independent of σ^2 in the same sense as (39): the noise contributes only through a σ^2 prefactor that is dominated by the $1/n$ scaling once $n > m\sigma^2/(\mu \cdot \text{target})$.

H.8 Discrete Sobolev target preservation

The Sobolev regularizer in Section 3.4 adds a gradient-domain term to the training loss. We show that this does not shift the population minimizer away from $f^\star$, and we extend the result to the geometry-weighted form actually used in Section 3.4.

Let fields be represented on a fixed grid and let D be a fixed linear finite-difference gradient operator. For $\alpha > 0$, the discrete Sobolev seminorm is $\|v\|^2_{H^1_\alpha} := \|v\|_2^2 + \alpha\|Dv\|_2^2$.

Proposition 7 (Discrete Sobolev population minimizer)**.** *Under (A1), for every measurable $f : \mathcal{X} \to \mathcal{Y}$,*

$$\mathbb{E}\|f(X) - Y\|^2_{H^1_\alpha} = \mathbb{E}\|f(X) - f^\star(X)\|^2_{H^1_\alpha} + \mathbb{E}\|\varepsilon\|^2_{H^1_\alpha}, \tag{49}$$

so the population minimizer of the discrete Sobolev loss is $f^\star$.

Proof. Define the linear map $A_0 v := (v, \sqrt{\alpha}\, Dv)$, so that $\|v\|^2_{H^1_\alpha} = \|A_0 v\|_2^2$. Then $A_0(f(X)-Y) = A_0(f(X) - f^\star(X)) - A_0\varepsilon$. Expanding the square and taking expectations,

$$\mathbb{E}\|f(X) - Y\|^2_{H^1_\alpha} = \mathbb{E}\|f(X) - f^\star(X)\|^2_{H^1_\alpha} - 2\,\mathbb{E}\langle A_0(f(X) - f^\star(X)),\, A_0\varepsilon\rangle + \mathbb{E}\|\varepsilon\|^2_{H^1_\alpha}. \tag{50}$$

Because A_0 is fixed and linear, it commutes with conditional expectation: $\mathbb{E}[A_0\varepsilon \mid X] = A_0\,\mathbb{E}[\varepsilon \mid X] = 0$ by (A1). The cross-term therefore vanishes by the tower property. □

Corollary 8 (Weighted Sobolev with X-measurable weights)**.** *Let $W(X) : \mathcal{Y} \to \mathcal{Y}$ be a $\sigma(X)$-measurable bounded linear operator on $\mathcal{Y}$ (e.g. pointwise multiplication by a bounded geometry-derived field $w_{\mathrm{dist}}(X)$). Then under (A1), the weighted Sobolev loss*

$$\mathcal{L}_W(f) := \mathbb{E}\|W(X)(f(X) - Y)\|^2_{H^1_\alpha} \tag{51}$$

satisfies $\mathcal{L}_W(f) = \mathbb{E}\|W(X)(f(X) - f^\star(X))\|^2_{H^1_\alpha} + \mathbb{E}\|W(X)\varepsilon\|^2_{H^1_\alpha}$, so its population minimizer (in the kernel-quotient sense determined by $W(X)$) is consistent with $f^\star$. In particular, the distance-modulated Sobolev loss in Section 3.4 does not bias the population target.

Proof. $W(X)A_0$ is $\sigma(X)$-measurable and linear, so $\mathbb{E}[W(X)A_0\varepsilon \mid X] = W(X)A_0\,\mathbb{E}[\varepsilon \mid X] = 0$ by (A1). The same expansion as in Proposition 7 eliminates the cross-term. □

The same argument applied with $\alpha = 0$ and $W(X)$ a geometry-confidence weight covers the confidence-modulated L^2 stage losses $\mathcal{L}_k$ in Section 3.4.

H.9 Stage-wise losses inherit the denoising property

Theorem 4 is stated for a single target. RayNO instead supervises three cumulative outputs against three staged labels, with a shared encoder and a weighted Huber objective, so it is worth confirming that the argument carries over. It does, and the reason is simple: the prediction depends on the scene, not on the label draw.

Let $\bar{F}_k(X) := \mathbb{E}[Y_k \mid X]$ be the cumulative stage-k target in the training representation, including the fixed pre-training conversion, and write $Y_k = \bar{F}_k(X) + \varepsilon_k$ with $\mathbb{E}[\varepsilon_k \mid X] = 0$ by Proposition 1. For any bounded weight $w_k \geq 0$ derived from the input,

$$\mathbb{E}\sum_p w_k\big(f_{\theta,k} - Y_k\big)^2 = \mathbb{E}\sum_p w_k\big(f_{\theta,k} - \bar{F}_k\big)^2 + C_k, \qquad C_k := \mathbb{E}\sum_p w_k\,\varepsilon_k^2, \tag{52}$$

where the sum runs over receiver pixels p. The cross term vanishes because, conditioned on X, both $f_{\theta,k}$ and w_k are deterministic while ε_k is centered. Since C_k does not involve θ, finite-ray error raises the expected loss by a constant and leaves the minimizer where it was.

Three features of the architecture might be expected to disturb this, and none does. The shared encoder couples the stages through θ, but Eq. (52) holds for each k separately at any fixed θ, so summing over stages preserves it. The outputs are cumulative rather than independent, which again only matters through θ. The errors $\varepsilon_1, \varepsilon_2, \varepsilon_3$ are correlated, since the staged labels come from simulations of the same scene, but the identity uses only the conditional mean of each ε_k and never a product of two of them.

Two remaining gaps are closed elsewhere. The objective uses a Huber rather than a squared loss; measured against the higher-ray target, more than 99.98% of geometry-weighted Huber residuals fall in the quadratic region, so the objective behaves as a weighted squared loss over essentially all of the training signal. Proposition 7 and Corollary 8 extend the same argument to the squared Sobolev terms of Eq. (8).

H.10 Checking the noise model against the simulator

Assumption (A1) and Proposition 1 are statements about Sionna RT, not about our model, so they can be checked directly. This section reports that check. It is relevant because the residual between the label and the low-fidelity input visibly follows walls and shadow boundaries, which may appear to contradict the zero-mean assumption.

The two observations are consistent once the gap between the label and the low-fidelity input is split into two parts,

$$Y_{\text{IF-GT}} - u = \big(f^\star(X) - u\big) + \varepsilon. \tag{53}$$

The first part is the propagation structure that the low-fidelity simulation does not compute, so it follows the scene geometry by design. Only ε is simulation noise, and Assumption (A1) asks only that it has zero conditional mean, not that it is independent of X. Writing the finite-ray field at a fixed scene X as

$$P_{s,m}(X) = \frac{1}{m}\sum_{r=1}^{m} Z_{s,X}(\xi_r), \tag{54}$$

Proposition 1 gives conditional mean equal to the converged stage field and variance decaying as $1/m$. Independence here refers to the sampled rays *after* the scene is fixed. The error may still depend on the scene, and on the low-fidelity map, through its variance and its spatial correlation. The structured residual a reader sees is $Y_{\text{IF-GT}} - u = [f^\star(X) - u] + \varepsilon$; the bracketed term is the repeatable reflected, diffracted, and scattered structure missing from the LF map, and only ε is the theorem's noise variable.

Protocol. We drew 512 independently randomized 10^6-ray realizations per stage on fixed scenes, varying only the Sionna seed. The comparison reference is a pooled higher-ray field assembled from separate runs that share no realization with the 512, so the test is not self-referential. We then measured how the error of the running average decays as realizations accumulate.

Result. The averaging-error slopes lie between -0.493 and -0.502 on a log–log scale, against the -0.5 that an unbiased Monte Carlo estimator predicts. The averaged linear-power fields settle within 0.21 dB of the reference, the largest mean offset across predefined spatial groups is 0.049 dB, and the averages show no plateau over the range tested. A biased estimator would show the opposite signature: the decay would flatten at the level of the bias instead of continuing. These measurements support conditional centering while leaving scene-dependent variance and spatial correlation intact, which is all (A1) requires. The independent-seed averaging experiment of Appendix G.3 is a second view of the same property, measured in the metrics the paper reports.

Loss geometry. One detail matters for the weighted Huber objective of Section 3.4, since the constant-offset argument of Eq. (52) is stated for a squared loss. More than 99.98% of geometry-weighted Huber residuals fall in the quadratic region of the Huber function when measured against the higher-ray target, so the objective behaves as a weighted squared loss over essentially all of the training signal, and Proposition 7 with Corollary 8 covers the Sobolev terms.

NeurIPS Paper Checklist

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 state the paper's main contributions: the RayNO architecture, evaluation against baselines and noisy training labels, and the theoretical explanation for denoising without clean references. These claims are supported by the method, empirical results, ablations, and theoretical appendix, and the scope is limited to the stated indoor ray-tracing setting.

   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: Appendix A discusses intended use, foreseeable risks, and explicit scope limitations, including single-frequency evaluation, indoor-only scenes, single-transmitter settings, scalar RSS prediction, and reliance on ray-traced supervision. The paper also discusses sensitivity to out-of-distribution scene shifts and biased label noise.

   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 theoretical appendix states the notation and standing assumptions used in the analysis and provides complete proofs for the denoising-without-clean-references result, the Monte Carlo conditional-unbiasedness argument, the population decomposition lemma, and Sobolev target-preservation extensions.

   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: The paper specifies the simulation setting, train/test protocol, model inputs, training targets, baselines, metrics, ablations, curriculum, optimization details, and compute footprint. We also provide the complete implementation and dataset links in the supplemental material to reproduce the reported results.

   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: We provide an anonymized supplemental code archive containing the full RayNO implementation, training and evaluation scripts, baseline scripts, and reproduction instructions. Dataset links are also provided for the generated ray-tracing data used in the main experiments.

   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: The main text describes the experimental protocol and evaluation setting, while the appendix provides the full training details, including data splits, input construction, optimizer, learning-rate schedule, loss weights, batching, precision, random seed, baseline tuning protocol, and evaluation metrics.

   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: The extended results appendix reports mean performance with 95% paired bootstrap confidence intervals for the main metrics supporting the paper's claims. The confidence intervals are computed by resampling held-out test scenes/samples with a fixed number of bootstrap draws.

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: The appendix reports the compute footprint for RayNO training, including the GPU class, training duration, epoch count, batch size, and the fact that no multi-GPU or multi-node infrastructure is required. It also documents the baseline tuning and training protocol so that the compute needed for reproduced comparisons is clear.

   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 work uses synthetic ray-traced wireless scenes and does not involve human subjects, private user data, scraped personal data, or deployed decision-making systems. The submission and supplemental artifacts are anonymized and prepared in accordance with the NeurIPS Code of Ethics.

   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: The broader impact and limitations section discusses positive impacts such as faster indoor network planning, improved coverage, and lower deployment cost. It also discusses risks, especially over-trust in surrogate predictions outside the evaluated regime, and states the intended operating scope and limitations.

    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 released artifacts consist of synthetic ray-tracing datasets, training/evaluation code, and neural surrogate models for indoor wireless field prediction. They do not include scraped personal data, human-generated content, pretrained generative models, or other assets with a high risk of direct misuse.

    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: The paper cites the existing simulation tools, software libraries, and baseline methods used in the experiments, including the ray-tracing simulator and neural-operator/baseline implementations. The supplemental material documents the relevant dependencies, versions, and license information, and all released code and data are distributed consistently with those terms.

    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: We release new synthetic ray-tracing datasets, code, and reproduction scripts as part of the supplemental material. The documentation describes the dataset structure, generation protocol, train/test splits, expected file formats, model checkpoints or scripts, and instructions for reproducing the main experiments.

    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, user studies, human-subject experiments, or human-generated annotations. All data are generated synthetically through controlled ray-tracing simulations.

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 research does not involve human subjects, crowdsourcing, personal data collection, behavioral data, or user-facing studies. Therefore, IRB or equivalent human-subjects review is not applicable.

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: LLMs are not used as an important, original, or non-standard component of the proposed method, experiments, data generation, or scientific claims. Any use of language models was limited to ordinary writing, editing, or formatting assistance and did not affect the core methodology or results.

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.