Can NeRFs "See" without Cameras?

Chaitanya Amballa[1] amballa2@illinois.edu
Sattwik Basu[1]* sattwik2@illinois.edu
Yu-Lin Wei[1]* yulinlw2@illinois.edu
Zhijian Yang[2] yzhijian@amazon.com
Mehmet Ergezer[2] mergezer@amazon.com
Romit Roy Choudhury[1,2] croy@illinois.edu

[1]University of Illinois Urbana–Champaign [2]Amazon

Abstract

Neural Radiance Fields (NeRFs) have been remarkably successful at synthesizing novel views of 3D scenes by optimizing a volumetric scene function. This scene function models how optical rays bring color information from a 3D object to the camera pixels. Radio frequency (RF) or audio signals can also be viewed as a vehicle for delivering information about the environment to a sensor. However, unlike camera pixels, an RF/audio sensor receives a mixture of signals that contain many environmental reflections (also called "multipath"). Is it still possible to infer the environment using such multipath signals? We show that with redesign, NeRFs can be taught to learn from multipath signals, and thereby "see" the environment. As a grounding application, we aim to infer the indoor floorplan of a home from sparse WiFi measurements made at multiple locations inside the home. Although a difficult inverse problem, our implicitly learnt floorplans look promising, and enables forward applications, such as indoor signal prediction and basic ray tracing.

1 Introduction

NeRFs [25, 43, 2, 45] have delivered impressive results in solving inverse problems, resulting in 3D scene rendering. While NeRFs have mostly used pictures (from cameras or LIDARs) to infer a 3D scene, we ask if the core ideas can generalize to the case of wireless signals (such as RF or audio). Unlike camera pixels that receive line-of-sight (LoS) rays, a wireless receiver (e.g., a WiFi antenna on a smartphone) would receive a mixture of LoS and many reflections, called *multipath*. If the receiver moves, it receives a sequence of N measurements. Using these N wireless measurements, is it possible to learn a representation of the scene, such as the floorplan of the user's home? A vanilla NeRF understandably fails since it is not equipped to handle multipath. This paper is focused on redesigning NeRFs so they can learn to image the environment, thereby solving the inverse problem from ambient wireless signals.

A growing body of work [3, 26, 47, 22, 23, 19] is investigating connections between NeRFs and wireless. While none have concentrated on imaging, NeRF2 [47] and NeWRF [22] have augmented NeRFs to correctly synthesize WiFi signals at different locations inside an indoor space. However, correct synthesis is possible without necessarily learning the correct signal propagation models. We find that NeRFs with adequate model complexity can overfit a function to correctly predict signals at test locations, but this function does not embed the true behavior of multipath signal propagation. We re-design the NeRF's objective function so that it learns the environment through line-of-sight (LoS) paths and reflections. This teaches the NeRF an implicit representation of the scene, which can then be utilized for various forward tasks, including WiFi signal prediction and ray tracing.

In our model, `EchoNeRF`, each voxel is parameterized by its opacity $\delta \in [0, 1]$ and orientation $\omega \in [-\pi, \pi]$. When trained perfectly, free-space air voxels should be transparent ($\delta = 0$), wall voxels should be opaque ($\delta = 1$), and each opaque voxel's orientation should match its wall's orientation.

*Equal contribution.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

 https://doi.org/10.52202/085713-3743

As measurements, we use the received *signal power*. Thus, the input to our EchoNeRF model is the transmitter (Tx) location, a sequence of known receiver (Rx) locations, and the signal power measured at each Rx location. The output of EchoNeRF is an (implicitly learnt) floorplan of the indoor space. We expect to visualize the floorplan by plotting the learnt voxel opacities.

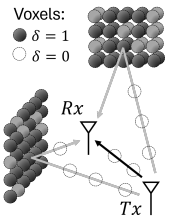

Figure 1: LoS and correct multi-path reflections.

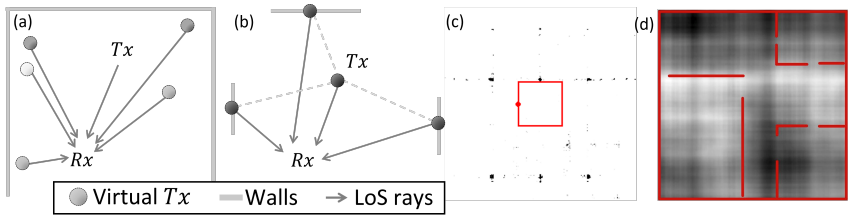

Figure 2: (a) Fitting signal power using virtual Txs. (b) Ideally, virtual Txs should be located on wall surfaces. (c) Sparse virtual Txs learnt by NeWRF shown in black/gray dots. (d) Dense virtual Txs learnt by NeRF2. Neither correspond to the true (red) floorplan.

Learning floorplans requires modeling the correct reflections (see Fig. 1) since these reflections help reveal where the walls are. However, without knowledge of the walls, the reflections are difficult to model, leading to a type of chicken and egg problem. Additionally, the number of wireless measurements is relatively sparse compared to the number of pixels measured in image-trained NeRFs. Finally, measured signals will have "blind spots", meaning that rays that bounced off certain regions of the walls may not have arrived at any of the Rx locations. This leaves gaps or holes in the floorplan and NeRF's interpolation through these gaps will produce error or blur.

EchoNeRF approaches this problem by modeling the received signal power as a combination of the LoS power and the power from first order reflections. The LoS model is inherited from classical NeRFs. The main departure from past work is in modeling the reflections. Since opaque voxels are unknown during training, the reflection surfaces are not known; hence, the reflection power at the Rx is modeled as an aggregate over all plausible reflections. Given the planar structure of walls, the plausible set of reflections can be heavily pruned to reduce the optimization complexity. Reflections aggregated over this plausible set models the total (LoS + reflection) power at a receiver Rx.

EchoNeRF trains to minimize the loss between the modeled and measured power across all Rx locations, and in the process, learns the voxel's opacities that best explain the measured dataset. Some regularization is necessary to cope with sparse measurements and to ensure smoothness of walls. Lastly, to handle some gradient imbalance issues, EchoNeRF freezes the LoS model once it converges, and uses this intermediate state to partly supervise the reflection model.

To evaluate EchoNeRF, we train on 2.4 GHz WiFi signals from NVIDIA's Sionna simulator [14], with floorplans from the Zillow's Indoor Dataset (ZIND) [9]. Results show consistent improvement over baselines in terms of the estimated floorplan's IoU and F1 score. Qualitative results show visually legible floorplans without any post-processing. Applying forward functions on the floorplan, EchoNeRF can predict the received signal power for new $\langle Tx, Rx \rangle$ locations (outperforming existing baselines). Lastly, basic ray tracing explains the predictions, offering interpretability to its results.

2 Related Work and Research Scope

Wireless (WiFi) channel prediction using NeRFs. NeWRF[22] and NeRF2[47] are recent papers that have used NeRFs to predict the wireless channel impulse response (CIR) [40] at unknown locations inside a room. Drawing a parallel to optical NeRFs, a voxel's color in optics becomes a voxel's transmit power in wireless. The voxel's density in optics remains the same in wireless, modeling how that voxel attenuates signals passing through it. NeRF2 and NeWRF *assign transmit power and attenuation to each voxel* such that they best explain the measured CIR. The authors explain that voxels assigned non-zero transmit power will be called *virtual transmitters*; these voxels represent the reflection points on the walls. However, *many assignments are possible that fit the CIR training data*, especially when the data is sparse. Fig. 2(a,b) illustrates 2 possible assignments. While the predicted CIRs could achieve low error for all such assignments, only one of the assignments will model the true reflections, forcing the virtual transmitters to be located on the wall surfaces (Fig. 2(b)). We have plotted NeRF2 and NeWRF's assignment of voxel densities (see Fig. 2(c,d)) to confirm

https://doi.org/10.52202/085713-3743

that the high accuracy in CIR prediction is not an outcome of correctly learning the wall layout. Our goal is to repair this important issue, i.e., assign voxel densities that obey the basic physics of wall reflections. Correct voxel assignment leads to the correct layouts, which then makes the (forward) CIR prediction easy.

Neural radiance fields for audio. Another active line of research focuses on predicting room impulse response (RIR) for audio [23, 37, 19, 6]. Neural Acoustic Field (NAF) [23] extended the classical NeRF to train on RIR measurements in a room and predict the RIR (magnitude and phase) at new $\langle Tx, Rx \rangle$ locations. NAF identified the possibility of overfitting to the RIR and proposed to learn, jointly, the local geometric features of the environment (as spatial latents) and the NAF parameters. The spatial latents embed floorplan information but a decoder needs to be trained using the partial floorplan data. `EchoNeRF` requires no floorplan supervision, and secondly, relies entirely on signal power (less informative than RIR) to estimate the floorplan.

Follow up work are embracing more information about the surroundings (pictures [20, 37, 19, 24], LIDAR scans [27], meshes and optical NeRFs [6]), to boost RIR accuracy. Results are steadily improving, however, this sequence of ideas is unaligned with solving the core inverse problem. Our goal is to first invert the signal power to a floorplan, which can then enable CIR/RIR predictions.

Modeling reflections in optical NeRFs Optical NeRFs have tackled reflections [10, 44, 33] for synthesizing glossy surfaces and mirrors, and for re-lighting [32, 39]. NeRFRen [12] proposes to decompose a viewed image into a transmitted and a reflected component. Ref-NeRF [41] also focuses on reflections through a similar decomposition of the transmitted and reflected color, however, the reflected color is modeled as a function of the viewing angle and the surface-normal, resulting in accurate models of specular reflection. Recent proposals such as NeRF-Casting [42] and oRCA [38] have further improved the models of multipath for glossy and mirrored surfaces, and others [5, 46] have developed similar ideas, with the core insights centering around solving a two-component decomposition problem. `EchoNeRF` faces the challenge of not knowing the number of rays adding up from all possible directions in the environment. Hence, `EchoNeRF` must solve a many-component decomposition problem by leveraging the physics of multipath signal propagation.
In the context of LIDARs, PlatoNeRF [18] and NeTF [34] cope with unknown reflections, however, LIDARs have very high time resolution (due to high clock frequency), and is therefore able to assign the incoming rays to different time buckets. This temporal separation allows the NeRF to make separate measurements for different surfaces in the scene. Since `EchoNeRF` uses only signal power—a single scalar measurement that contains a mixture of the LoS and all the reflections—the inverse problem of recovering the voxel attributes from only the power is far more complex.

3 `EchoNeRF` Model

Setup and Overview. At a Rx location, we model the received signal power ψ as

$$\psi = \psi_{LoS} + \psi_{ref_1} + \cdots + \psi_{ref_n}$$

where ψ_{LoS} is the power from the direct line-of-sight (LoS) path, and ψ_{ref_k} is the aggregate power from all k^{th} order reflections (i.e., all signal paths that underwent exactly k reflections before arriving at the Rx) [2] We assume M fixed transmitters and move the Rx to N known locations and measure ψ at each of them. `EchoNeRF` accepts $M \times N$ measurements as input and outputs the 2D floor-plan F, a binary matrix of size $L \times L$, where L denotes the maximum floorplan length.

We train `EchoNeRF` on the measured data using our proposed objective function. This function only models the LoS and the first order reflections. We disregard the higher orders since they are very complex to model and contribute, on average, $< 6\%$ of the total power (see statistics in Appendix C). The NeRF model we use is a remarkably simple MLP designed to predict the density $\delta \in [0, 1]$ and orientation $\omega \in [-\pi, \pi]$ of a specified voxel in the indoor scene. The orientation aids in modeling reflections. The proposed objective function – parameterized by voxel attributes $\langle \delta, \omega \rangle$ and the $\langle Tx, Rx \rangle$ locations – models an approximation of the received power ψ at that Rx location. Minimizing L_2 loss of this power across all Rx locations trains the MLP. Plotting out all the voxel densities in 2D gives us the estimated floorplan F.

[2] This model is a simplification since it ignores the signal phase in estimating the received power. Appendix B shows that with a moving wide-band receiver, like WiFi, the approximation may be tolerable for a sensing application like `EchoNeRF`.

 https://doi.org/10.52202/085713-3743

3.1 The LoS Model

Friss' equation [1] from electromagnetics models the free-space received power as $P_r = \frac{K}{d^2}$ where d is the distance of signal propagation, and K is a product of transmit-power, wavelength, and antenna-related constants [1]. We model this free-space (LoS) behavior in the NeRF framework through the following equation.

$$\psi_{LoS} = K \frac{\prod_{\{i|v_i \in LoS\}} (1 - \delta_i)}{d^2} \tag{1}$$

where K can be empirically measured, and d is the known distance between the $\langle Tx, Rx \rangle$. The numerator includes the product of voxel densities over all voxels along the LoS ray from Tx to Rx (with an abuse of notion, we write this as $v_i \in LoS$). This models occlusions. When the LoS path is completely free of any occlusions (i.e., $\delta_i = 0, \forall i$ where $\{i|v_i \in LoS\}$), we expect the received power to only be attenuated by the pathloss factor d^2 (in the denominator). Eq. 1 has a slight difference to classical NeRF's volumetric scene function. In our case, voxels along the ray do not contribute to the received power (whereas in NeRF, each voxel's color is aggregated to model the final pixel color at the image). In other words, we have modeled a single transmitter in Eq. 1.

3.2 The Reflection Model

To model reflections, consider a voxel v_j. Whether v_j reflects a ray from the Tx towards Rx depends on (1) v_j's density δ_j and orientation ω_j, (2) the $\langle Tx, Rx \rangle$ locations, and (4) whether the path from Tx to v_j, and from v_j to Rx are both occlusion-free. Parameterized by these, Eq. 2 models $\psi_{ref}(v_j)$, which is the received power at Rx due to the signal that reflected off voxel v_j.

$$\psi_{ref}(v_j) = \delta_j f(\theta, \beta) \frac{\prod_{k \in \{Rx:v_j\}} (1 - \delta_k) \prod_{l \in \{v_j:Tx\}} (1 - \delta_l)}{\left(d_{Tx:v_j} + d_{v_j:Rx}\right)^2} \tag{2}$$

Let us explain this equation briefly. The leading δ_j ensures that voxel v_j is not a reflector when $\delta_j = 0$. The $f(\theta, \beta)$ term models the wave-surface interactions, i.e., how signals get attenuated as a function of the incident angle θ and how signals scatter as a function of the offset angle β (which is the angle between the reflected ray and the direction of the Rx from v_j). The next two product terms ensure that for the Rx to receive this reflection, the voxels along the 2 segments (Tx to v_j and v_j to Rx) must be non-opaque; if any δ_k or δ_l equals 1, that reflection path is blocked, producing no power contribution via this voxel v_j to the receiver Rx. Finally, the denominator is the squared distance from Tx to v_j, and from v_j to Rx, modeling signal attenuation.

To compute the full reflection power, the natural question is: *which voxels are contributing to the received power?* Geometrically, any opaque voxel can be a plausible reflection point between any $\langle Tx, Rx \rangle$ pair. This is because, for triangle formed by Tx, v_j, and Rx, the voxel orientation ω_j can be assigned a direction that bisects the angle at v_j. For this ω_j, the reflected ray will perfectly arrive at the Rx. Thus, without the knowledge of orientation and density, the total first order reflection power at the Rx should be modeled as the sum of reflections on all voxels. This makes the optimization problem excessively under-determined.

We address this by modeling ω_j as a discrete value—multiples of $\frac{\pi}{K_\omega}$. Larger K_ω is needed when the environment has complex surface orientations; however, most floorplans exhibit perpendicular walls [48, 8, 11] and $K_\omega = 4$ is adequate. Once ω_j becomes discrete, the voxels that can produce plausible reflections become far fewer – we call this the "plausible set" $\mathcal{V}$. Fig. 3 visualizes $\mathcal{V}$ and shows 3 out of many plausible reflections from voxels with $\omega_j = 135°$. Eq. 3 sums up the power from all reflections that occur on the plausible set:

$$\psi_{ref_1} = \sum_{\{j|v_j \in \mathcal{V}\}} \psi_{ref}(v_j) \tag{3}$$

Thus, the final modeled power at a specific Rx location becomes $\tilde{\psi} = \psi_{LoS} + \psi_{ref_1}$.

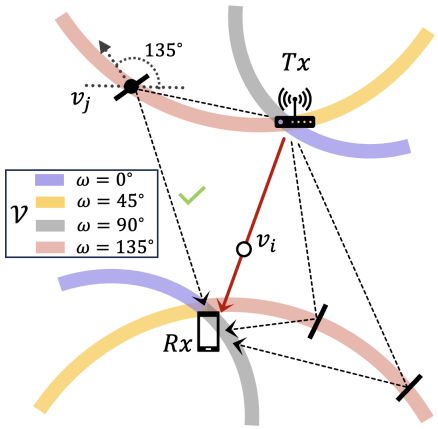

Figure 3: Colored stripes define the manifold from which reflections are plausible between $\langle Tx, Rx \rangle$. Voxels located on the manifold form the plausible set $\mathcal{V}$. Dashed lines show plausible reflections.

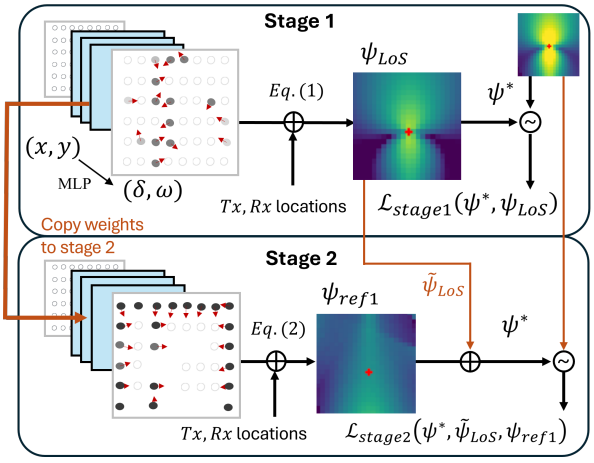

Figure 4: `EchoNeRF`'s two-stage training approach: In Stage 1, the LoS model is trained using known Rx locations and signal power. This provides a warm-start to the reflection model in Stage 2 which refines the learned voxel densities and orientation.

3.3 Gradient Issues during Training

Training against a L_2 loss, $\mathcal{L} = \|\tilde{\psi} - \psi^*\|_2^2$ [3], did not generate legible floorplans. We found that the ψ_{LoS} dominated the loss term, drowning the reflection model's influence on learning. At a high level, the gradient of the LoS model (Eq. 1) w.r.t. δ_i has fewer terms in the numerator's product, and a smaller d^2 in the denominator. The reflection model's gradient w.r.t. δ_i has many more terms since the reflection path is much longer; the denominator is also larger. Since $(1 - \delta_j) \leq 1$, their products force the gradient to decrease geometrically with more terms, causing the reflection gradient to be much smaller compared to LoS. We formalize this explanation below by considering the LoS and reflection losses individually[4].

$$\mathcal{L}_{LoS} = \left\| \tilde{\psi}_{LoS} - \psi^*_{LoS} \right\|_2^2, \quad \mathcal{L}_{ref} = \left\| \tilde{\psi}_{ref} - \psi^*_{ref} \right\|_2^2$$

Consider the gradient of $\mathcal{L}_{LoS}$ w.r.t the density of v_i.

$$\nabla_{\delta_i} \mathcal{L}_{LoS} = 2(\tilde{\psi}_{LoS} - \psi^*_{LoS}) \sum_{\{n | i \in LoS^{(n)}\}} \nabla_{\delta_i} \psi^{(n)}_{LoS}$$

$$\text{where } \nabla_{\delta_i} \psi^{(n)}_{LoS} = -\frac{K}{{d^{(n)}}^2} \prod_{\substack{j \in LoS^{(n)} \\ j \neq i}} (1 - \delta_j) \tag{4}$$

where $LoS^{(n)}$ is the n-th LoS path passing through voxel v_i.

The gradient of $\mathcal{L}_{ref}$ has a nearly identical expression with the only difference being many more product terms and that $\nabla_{\delta_i} \psi^{(n)}_{ref}$ depends on where v_i is present in the n-th set of reflection path voxels (denoted by $Ref^{(n)}$). For example,

$$\nabla_{\delta_i} \psi^{(n),Tx}_{ref} = -C \prod_{\substack{j \in Ref^{(n)}_{Tx:v} \\ j \neq i}} (1 - \delta_j) \prod_{k \in Ref^{(n)}_{v:Rx}} (1 - \delta_k) \tag{5}$$

$$C = \frac{\delta^{(n)} f^{(n)}(\theta, \beta)}{(d^{(n)}_{Tx:v} + d^{(n)}_{v:Rx})^2}$$

[3] Here, ψ^* denotes the ground truth signal power

[4] For the ease of explanation, we use ψ^*_{LoS} and ψ^*_{ref} to denote the ground truth LoS and reflection powers, respectively. We do not need to know these terms in practice.

https://doi.org/10.52202/085713-3743

here $\nabla_{\delta_i} \psi_{ref}^{(n),Tx}$ denotes the gradient when v_i is between *Tx* to the reflection point v.

Finally, since the modeled power approximates the measured power, the residual error will remain non-zero even if the floorplan is accurately learnt. As a result, the optimization is biased towards voxels of higher gradients, i.e., voxels on the LoS path, suppressing the importance of reflections. To address this, we train `EchoNeRF` in 2 stages.

3.4 Multi-stage Training

Stage 1: We first use the LoS model against the measured ground truth power ψ^* (see Fig. 4). This converges quickly because the network easily learns the transparent voxels (δ=0) located along LoS paths. For LoS paths that are occluded, the network *incorrectly* learns excessive opaque voxels between the $\langle Tx, Rx \rangle$, but this does not affect the LoS error since the path is anyway occluded. Hence, the outcome is a crude floorplan but a near-perfect LoS power estimate $\tilde{\psi}_{LoS}$. We utilize this $\tilde{\psi}_{LoS}$ in stage 2 (discussed soon).

As Stage 1 training progresses, some opaque voxels emerge, offering crude contours of some walls. We estimate a voxel's spatial gradient, $\nabla\delta_i$, and use it to supervise the orientation ω_i of that voxel. The intuition is that a voxel's orientation – needed to model reflections in Stage 2 – is essentially determined by the local surface around that voxel. The gradient $\nabla\delta_i$ offers an opportunity for weak supervision. Thus, the loss for Stage 1 is:

$$\mathcal{L}_{stage1} = \|\psi^* - \psi_{LoS}\|_2^2 + \mathcal{L}^+ \tag{6}$$

$$\text{where } \mathcal{L}^+ = \lambda_1 \sum_{\forall j} \|\nabla\delta_j - \omega_j\|_2^2 + \lambda_2 \mathcal{L}_{reg} \tag{7}$$

with $\lambda_1, \lambda_2 > 0$ being tunable hyperparameters. The regularization term $\mathcal{L}_{reg}$ will be discussed soon. Finally, the near-perfect estimate of LoS power, denoted $\tilde{\psi}_{LoS}$, is also carried over to Stage 2 to ensure the reflection model is penalized when it veers away from this LoS estimate.

Stage 2 focuses on training the reflection model using the following loss function.

$$\mathcal{L}_{stage2} = \left\|\tilde{\psi}_{LoS} - \psi_{LoS}\right\|_2^2 + \left\|\psi^* - \tilde{\psi}_{LoS} - \psi_{ref_1}\right\|_2^2 + \mathcal{L}^+$$

The first term in the RHS ensures that the Stage 1's LoS estimate is honored in Stage 2. The second term subtracts Stage 1's LoS power from the measured power, ($\psi^* - \tilde{\psi}_{LoS}$); this models the total power *only due to reflections*. Our (first order) reflection model ψ_{ref_1} is trained to match this aggregate power (L_2 loss). The supervision on orientation and the regularization terms are the same as in Stage 1.

■ **Regularization**: Floorplans demonstrate significant local similarity in orientation, hence we penalize differences in orientation among neighbors, using a regularization (Eq. 8) similar to Total Variation [31]. This can be achieved without additional computational cost to the neural network by directly utilizing voxel orientations obtained from each ray.

$$\mathcal{L}_{\text{reg}} = \frac{1}{n_v(n_r - 1)} \sum_{n=1}^{n_v} \sum_{i=1}^{n_r - 1} \|\omega_{n,i+1} - \omega_{n,i}\|_2^2 \tag{8}$$

Here n_v is the number of voxels queried from the plausible set $\mathcal{V}$ and n_r is the number of voxels along the each ray.

4 Experiments

Floorplan and Wireless Simulation Dataset. Floorplans are drawn from the Zillow Indoor Dataset [9]. In each floorplan, we use the A^* algorithm [13] to generate a walking trajectory that traverses all rooms. We use the NVIDIA Sionna RT [15, 14] – a ray tracer for radio propagation modeling – to compute the ground truth signal power (also known as *received signal strength index* (RSSI)). We randomly place M *Tx*s, one in each room, denoted as T_m. To simulate omnidirectional transmissions

at 2.4GHz from each *Tx* location, we shoot 10^7 rays into the given floorplan. For receiver locations, we sample the user trajectory at a fixed time interval to obtain N *Rx* locations, denoted as R_n. Sionna accounts for specular reflections and refraction when these rays interact with walls in the specified floor plan; we use the default materials for the walls. As with most WiFi simulators, Sionna does not model signal penetration through walls – this means that a *Tx* and *Rx* located on opposite sides of a wall will not receive any RSSI. Overall, we gather $M \times N$ RSSI measurements $(T_m, R_n, \psi_{m,n})$ that serve as input to `EchoNeRF`.

Baselines used for comparison are:

1. `NeRF2` [47]: Models WiFi reflections via virtual transmitters to predict channel impulse response (CIR).
2. `Heatmap Segmentation` [21]: Interpolates CIR across the whole floorplan and applies an image segmentation algorithm (on the interpolated RSSI heatmap) to isolate each room. Essentially, the algorithm identifies the contours of sharp RSSI change since such contours are likely to correspond to walls. Implementation details are included in the Appendix E.4.
3. `MLP`: Trains an MLP network to directly estimate the RSSI based on *Tx* and *Rx* locations.
4. `EchoNeRF_LoS`: Reports `EchoNeRF`'s result considering only LoS path (ablation study).

Metrics. We evaluate using 3 metrics:

(A) Wall Intersection over Union (Wall_IoU): This metric measures the degree to which the predicted walls and the true walls superimpose over each other in the 2D floorplan. The following equation defines the metric:

$$\texttt{Wall\_IoU} = \frac{WP \cap WP^*}{WP \cup WP^*}$$

where WP denotes the set of predicted wall pixels and WP^* denotes the true wall pixels. This is a *harsh metric* given wall pixels are a small fraction of the total floorplan; if a predicted wall is even offset by one pixel from the true wall, the `Wall_IoU` drops significantly. IoU [30] has often been defined in terms of room pixels (instead of wall pixels); this is an overestimate in our opinion, since predicting even an empty floorplan results in an impressively high IoU.

(B) F1 score [35]: Defined as $F1 = \frac{2 \times P \times R}{P+R}$, where P is the *precision* and R is the *recall* of the bitmap. P and R are defined based on wall pixels, similar as above.

(C) RSSI Prediction Error (RPE): We split all Rx locations into a training and test set. RPE reports the average median RSSI error over all the test locations across floorplans.

4.1 Overall Summarized Results

	2000 receiver locations			1000 receiver locations		
Method	Wall_IoU ↑	F1 Score ↑	RPE ↓	Wall_IoU ↑	F1 Score ↑	RPE ↓
`MLP`	-	-	1.03	-	-	0.65
`Heatmap Seg.`	0.12 ± 0.03	0.21 ± 0.05	1.32	0.09 ± 0.02	0.16 ± 0.04	1.46
`NeRF2`	0.14 ± 0.02	0.24 ± 0.03	4.36	0.12 ± 0.02	0.21 ± 0.04	4.2
`EchoNeRF_LoS`	0.27 ± 0.07	0.42 ± 0.10	9.12	0.25 ± 0.04	0.39 ± 0.06	10.86
`EchoNeRF`	0.38 ± 0.06	0.55 ± 0.06	3.56	0.32 ± 0.06	0.48 ± 0.05	4.32

Table 1: Performance Results for Wall_IoU, F1 Score, and RPE

Table 1 reports comparative results between `EchoNeRF` and baselines, averaged over 20 different experiments, using all 3 metrics. The number of measurements are sparse ($N = 2000$ and $N = 1000$), given that apartment sizes in our dataset are more than $250,000$ pixels. Mean and standard deviation are reported in the table. `EchoNeRF` outperforms all models in terms of `Wall_IoU` and `F1 Score`. Compared to `EchoNeRF_LoS`, `EchoNeRF` demonstrates visible improvements, highlighting the advantage of modeling reflections. The absolute `Wall_IoU` values are understandably low because *the metric penalizes small errors.*

`NeRF2` is unable to predict the floor plan (opaque voxels) well and is only able to achieve better RPE than `EchoNeRF_LoS`. `EchoNeRF` outperforms both `EchoNeRF_LoS` and `NeRF2`. Interestingly, MLP incurs a lower RPE than `NeRF2` suggesting that RSSI is amenable to interpolation, and `NeRF2`'s implicit representation may not be an advantage for this interpolation task.

 https://doi.org/10.52202/085713-3743

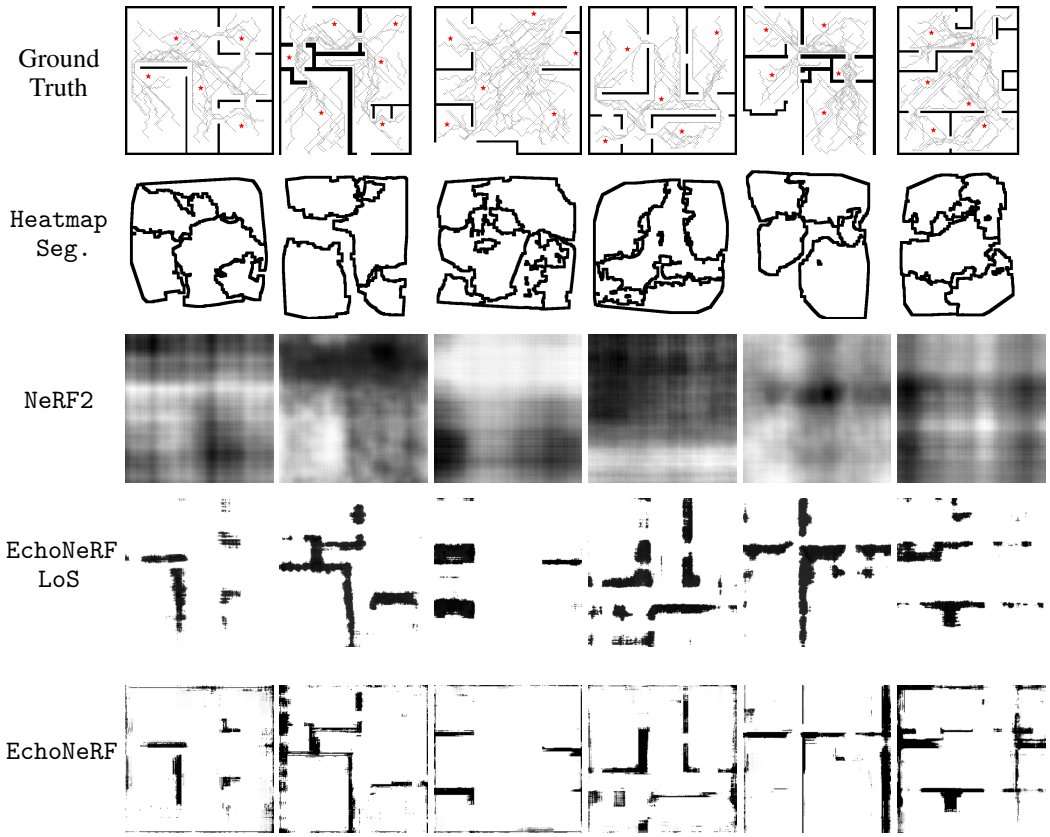

Figure 5: Qualitative comparison of ground truth floorplans against baselines. In the first row, red stars denote *Tx* locations and light gray dots denote *Rx* measurement locations. The bottom two rows show floorplans learnt by `EchoNeRF_LoS` (i.e., Stage 1) and `EchoNeRF` (i.e., Stage 2) with sharper walls and boundaries. More visualizations available at `https://echonerf.github.io/`

4.2 Qualitative Results: Visual floorplans, RSSI heatmap, and basic ray tracing

■ **Visual floorplans.** Figure 5 presents visualization from all baselines and a comparison with our LoS-only model (as ablation). All the floorplans use $N = 2000$ receiver locations. We make the following observations. (1) `Heatmap Segmentation` leverages the difference of RSSI on opposite sides of a wall, however, reflections pollute this pattern, especially at larger distances between Tx and Rx. Further, signals leak through open doors, injecting errors in the room boundaries. (2) `NeRF2` performs poorly since its MLP learns one among many possible assignments of virtual transmitters to fit the RSSI training data. The virtual transmitters hardly correlate to the walls of the environment. (3) `EchoNeRF_LoS` can infer the position of inner walls. However, these walls are thick and slanted because while `EchoNeRF_LoS` can identify occlusions between a $\langle Tx, Rx \rangle$ pair, it cannot tell the shape and pattern of these occlusions. Crucially, `EchoNeRF_LoS` also cannot infer the boundary walls since no receivers are located outside the house. (4) `EchoNeRF` outperforms the baselines, sharpens the inner walls compared to `EchoNeRF_LoS`, and constructs the boundary walls well.

Shortcomings: Recall that some parts of the floorplan are in the "blind spots" of our dataset since no reflection arrives from those parts to any of our sparse Rx locations (e.g., see bottom left corner of the 1^{st} floorplan; no signals reflect off this region to arrive at any of the Rx locations). Hence, `EchoNeRF` is unable to construct the bottom of the left wall in this floorplan. Finally, note that areas outside the floorplan (e.g., the regions on the right of 6^{th} floorplan) cannot be estimated correctly since no measurements are available from those regions (hence, those voxels do not influence the gradients).

■ **RSSI prediction.** Figure 6 visualizes and compares predicted RSSI. The top row shows predictions at new Rx locations with the Tx held at the trained location; the bottom row shows predictions when

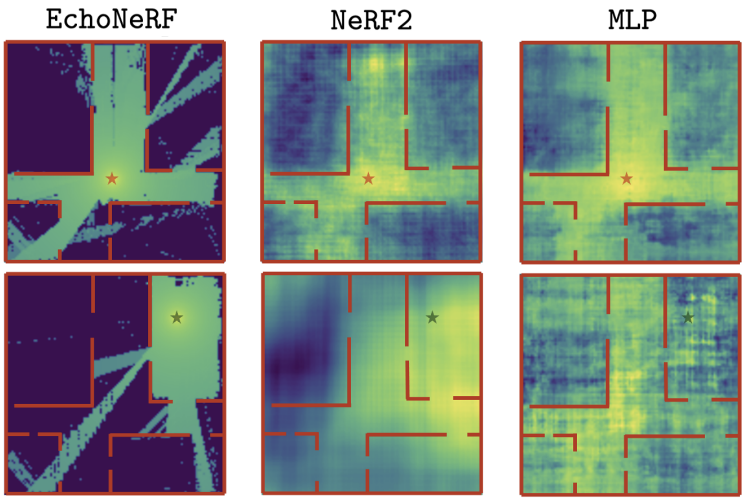

Figure 6: Heatmaps highlighting EchoNeRF's ability to learn signal propagation. (Top row) Inferred RSSI heatmaps with Tx (red star) as used in training. (Bottom row) A new Tx (green star) degrades NeRF2 and MLP while EchoNeRF shows accurate predictions.

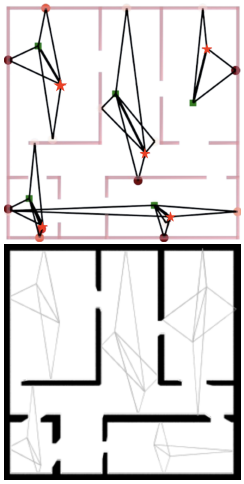

Figure 7: (a) Tracing reflections on the learnt floorplan. (b) True reflections from Sionna.

both Tx and Rx are moved to new locations. Two key observations emerge: (1) EchoNeRF is limited by Sionna's inability to simulate through-wall signal penetration; NeRF2 has access to an expensive license for a through-wall simulation and shows better predictions inside the rooms. However, in areas that EchoNeRF can "see" (e.g., corridors in the top row), the awareness of reflecting surfaces leads to significantly better predictions. (2) When the Tx location differs from that used in training, EchoNeRF's improvement over NeRF2 is significant. This is the core advantage of first solving the inverse problem and then leveraging it for the (forward) RSSI prediction.

■ **Learning reflected rays.** For a given $\langle Tx, Rx \rangle$ pair, we examine the points in the plausible set $\mathcal{V}$ that contribute to the reflections. Fig. 7 compares the ray-tracing results from the NVIDIA Sionna simulator (we pick only first order reflections). EchoNeRF captures many of the correct reflections. Of course, some are incorrect – a false positive occurs in the bottom right room since some wall segment is missing in our estimate; false negatives also occur in the top right room where again some parts of the wall are missing.

4.3 Relaxing Assumptions & Sensitivity Study

■ **Transmitter's location.** We assumed knowledge of Tx locations, however, we relax this by applying maximum likelihood estimation on observed RSSI power, ψ^* (see Appendix D). On average, the estimated *Tx* location error is 2.08 pixels in floorplans of sizes $\approx 512 \times 512$ pixels.

■ **Receiver location error.** Table 2 shows EchoNeRF's sensitivity to Rx location errors. We inject Gaussian noise $\mathcal{N}(0, \sigma^2 I)$ to the Rx locations; $\sigma = 1$ implies a physical error of 1m. Wall_IoU accuracy obviously drops with error but 0.5 meter of error is tolerable without destroying the floorplan structure. Advancements in WiFi positioning systems have demonstrated robust sub-meter error.

Error σ(m)	0	0.5	1	2
Wall_IoU	0.38	0.35	0.33	0.29

Table 2: Estimated Wall_IoU at various levels of injected noise σ

Figure 8: EchoNeRF's floorplan inference with furniture in conference (left) and apartment (right) layouts.

■ **Effect of Furniture.** Fig. 8 visualizes inferred floorplans when toy objects are scattered in open spaces (Rx locations remain N=2000). EchoNeRF is able to identify some of the object blobs but sharpening the small objects is challenging due to more higher order reflections from furniture. Follow up work is needed, either in modeling 2^{nd} order reflections or by imposing stronger regularizations.

https://doi.org/10.52202/085713-3743

Material	EchoNeRF_LoS↑	EchoNeRF↑
Concrete	0.251	0.371
Glass	0.236	0.364
Brick	0.232	0.357
Marble	0.226	0.328
Wood	0.227	0.311

Table 3: Sensitivity across materials.

SNR (dB)	EchoNeRF_LoS↑	EchoNeRF↑	Qualitative
∞ (no noise)	0.251	0.371	Legible
60	0.246	0.336	Legible
50	0.231	0.292	Legible
40	0.226	0.298	Legible
30	0.207	0.241	Missing walls
10	0.090	0.140	Illegible

Table 4: Noise robustness across SNR levels.

■ **Material Sensitivity.** Table 3 shows the mean Wall_IoU of EchoNeRF on five different materials averaged across six scenes (shown in Fig. 5) Materials with higher reflectivity, such as concrete and glass, yield better performance than absorptive materials like wood. This is because more reflections allow better performance for EchoNeRFś reflection model.

■ **Robustness to RSSI error.** We added Gaussian noise to the RSSI measurements with a mean equal to the noise floor (in dB) and a variance of 4 dB. We vary the noise floor levels ranging from -80 dB to -130 dB across the 6 floorplans shown in Fig. 5. The SNR at a receiver is computed as the difference between the received signal power and the noise floor (e.g., a received power of -70 dB with a noise floor of -80 dB results in an SNR of 10 dB). We report the mean Wall_IoU for EchoNeRF_LoS and EchoNeRF in Table 4. The performance drops with decreasing SNR; both EchoNeRF_LoS and EchoNeRF's floorplans are still legible till 30 dB, but below that (when noise power becomes comparable to the signal power) they break down, leading to missing and illegible walls. Our results are conservative; when we report a specific SNR level (e.g., 40dB), it represents the highest SNR (best-case) among all receivers in that scenario, meaning other receivers experience even lower SNRs. In practice, WiFi SNR ranges from 30-60dB depending on the distance from the router, with close-proximity measurements often exceeding 50-60dB. For average real-world SNR conditions around 45dB, the corresponding best-case SNR would be 60+dB, which aligns with the top rows where EchoNeRF demonstrates strong performance.

5 Follow ups and Conclusion

Follow-ups. (1) The ability to model 2^{nd} order reflections will boost EchoNeRF's accuracy, allowing it to sharpen the scene and decode smaller objects. For short range applications, such as non-intrusive medical imaging, 2^{nd} and 3^{rd} order reflections would be crucial. This remains an important direction for follow-on research. (2) Extending EchoNeRF to 3D floorplans is also of interest, and since it is undesirable to increase the number of measurements, effective 3D priors, or 2D-to-3D post-processing, may be necessary. Such post-processing tools exist [7] but we have not applied them since our goal is to improve NeRF's inherent inverse solver. (3) Expanding evaluations beyond ZInD, which contains largely unfurnished, rectangular rooms, to richer datasets such as HM3D [29] and MVL [36] would also be valuable: furniture and clutter can introduce significant multipath effects that complicate RF signal modeling, and additional research is needed to understand these effects. We defer this to future work, as our focus in this paper is not so much to understand the limits of RF-based NeRFs, but to establish the feasibility of such frameworks. (4) Finally, EchoNeRF floorplans can offer valuable spatial context to Neural RIR synthesizers like [47, 22, 6, 19, 20]. Synthesized RIR could in-turn aid EchoNeRF's floorplan inference, forming the basis for an alternating optimization strategy. We leave these ideas to follow-up research.

Conclusion. In summary, we re-design the NeRF framework so it can learn to "see" its environment by leveraging both line-of-sight (LoS) paths and multipath reflections. While such reflections bring to the sensor more information about the surroundings, their mixtures with the LoS path also complicates the core inverse problems. EchoNeRF takes a step towards solving this inverse problem, but also leaves room for further research in neural wireless imaging and varies downstream applications.

Acknowledgments

We would like to thank Prof. Shenlong Wang for his guidance in the initial phase of this work. We also thank the anonymous reviewers for their valuable feedback. This work was partially supported by NSF #2008338, #1909568, #2148583, and #MRI-2018966. This work used DELTA at NCSA through allocation CIS230230 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by U.S. National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296

References

[1] Constantine A. Balanis. *Antenna Theory: Analysis and Design*. John Wiley & Sons, 3rd edition, 2016.

[2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5855–5864, 2021.

[3] Amartya Basu and Ayon Chakraborty. Specnerf: Neural radiance field driven wireless coverage mapping for 5g networks. In *Proceedings of the Twenty-Fifth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, MobiHoc '24, page 440–445, New York, NY, USA, 2024. Association for Computing Machinery.

[4] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[5] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12684–12694, 2021.

[6] Amandine Brunetto, Sascha Hornauer, and Fabien Moutarde. Neraf: 3d scene infused neural radiance and acoustic fields. *arXiv preprint arXiv:2405.18213*, 2024.

[7] Cedreo. Convert 2d floor plan to 3d, 2025. Accessed: 2025-03-01.

[8] James M Coughlan and Alan L Yuille. Manhattan world: Compass direction from a single image by bayesian inference. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 941–947. IEEE, 1999.

[9] Steve Cruz, Will Hutchcroft, Yuguang Li, Naji Khosravan, Ivaylo Boyadzhiev, and Sing Bing Kang. Zillow indoor dataset: Annotated floor plans with 360deg panoramas and 3d room layouts. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2133–2143, 2021.

[10] Wenhang Ge, Tao Hu, Haoyu Zhao, Shu Liu, and Ying-Cong Chen. Ref-neus: Ambiguity-reduced neural implicit surface learning for multi-view reconstruction with reflection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4251–4260, 2023.

[11] Haoyu Guo, Sida Peng, Haotong Lin, Qianqian Wang, Guofeng Zhang, Hujun Bao, and Xiaowei Zhou. Neural 3d scene reconstruction with the manhattan-world assumption. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5511–5520, June 2022.

[12] Yuan-Chen Guo, Di Kang, Linchao Bao, Yu He, and Song-Hai Zhang. Nerfren: Neural radiance fields with reflections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18409–18418, June 2022.

[13] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[14] Jakob Hoydis, Fayçal Aït Aoudia, Sebastian Cammerer, Merlin Nimier-David, Nikolaus Binder, Guillermo Marcus, and Alexander Keller. Sionna rt: Differentiable ray tracing for radio propagation modeling. In *2023 IEEE Globecom Workshops (GC Wkshps)*, pages 317–321. IEEE, 2023.

[15] Jakob Hoydis, Sebastian Cammerer, Fayçal Ait Aoudia, Avinash Vem, Nikolaus Binder, Guillermo Marcus, and Alexander Keller. Sionna: An open-source library for next-generation physical layer research. *arXiv preprint arXiv:2203.11854*, 2022.

 https://doi.org/10.52202/085713-3743

[16] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[18] Tzofi Klinghoffer, Xiaoyu Xiang, Siddharth Somasundaram, Yuchen Fan, Christian Richardt, Ramesh Raskar, and Rakesh Ranjan. Platonerf: 3d reconstruction in plato's cave via single-view two-bounce lidar. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14565–14574, 2024.

[19] Susan Liang, Chao Huang, Yapeng Tian, Anurag Kumar, and Chenliang Xu. Av-nerf: Learning neural fields for real-world audio-visual scene synthesis. *Advances in Neural Information Processing Systems*, 36:37472–37490, 2023.

[20] Susan Liang, Chao Huang, Yapeng Tian, Anurag Kumar, and Chenliang Xu. Neural acoustic context field: Rendering realistic room impulse response with neural fields. *arXiv preprint arXiv:2309.15977*, 2023.

[21] Dingding Liu, Bilge Soran, Gregg Petrie, and Linda Shapiro. A review of computer vision segmentation algorithms. *Lecture notes*, 53, 2012.

[22] Haofan Lu, Christopher Vattheuer, Baharan Mirzasoleiman, and Omid Abari. Newrf: A deep learning framework for wireless radiation field reconstruction and channel prediction. In *Forty-first International Conference on Machine Learning*, 2024.

[23] Andrew Luo, Yilun Du, Michael Tarr, Josh Tenenbaum, Antonio Torralba, and Chuang Gan. Learning neural acoustic fields. *Advances in Neural Information Processing Systems*, 35:3165–3177, 2022.

[24] Sagnik Majumder, Changan Chen, Ziad Al-Halah, and Kristen Grauman. Few-shot audio-visual learning of environment acoustics. *Advances in Neural Information Processing Systems*, 35:2522–2536, 2022.

[25] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[26] Tribhuvanesh Orekondy, Pratik Kumar, Shreya Kadambi, Hao Ye, Joseph Soriaga, and Arash Behboodi. WineRT: Towards neural ray tracing for wireless channel modelling and differentiable simulations. In *The Eleventh International Conference on Learning Representations*, 2023.

[27] Zainab Oufqir, Abdellatif El Abderrahmani, and Khalid Satori. Arkit and arcore in serve to augmented reality. In *2020 International Conference on Intelligent Systems and Computer Vision (ISCV)*, pages 1–7. IEEE, 2020.

[28] Paolo Prandoni and Martin Vetterli. *Signal Processing for Communications*. CRC Press, Boca Raton, FL, 2008.

[29] Santhosh K Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, et al. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. *arXiv preprint arXiv:2109.08238*, 2021.

[30] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression, 2019.

[31] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268, 1992.

[32] Viktor Rudnev, Mohamed Elgharib, William Smith, Lingjie Liu, Vladislav Golyanik, and Christian Theobalt. Nerf for outdoor scene relighting. In *European Conference on Computer Vision*, pages 615–631. Springer, 2022.

[33] Seunghyeon Seo, Yeonjin Chang, and Nojun Kwak. Flipnerf: Flipped reflection rays for few-shot novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22883–22893, 2023.

[34] Siyuan Shen, Zi Wang, Ping Liu, Zhengqing Pan, Ruiqian Li, Tian Gao, Shiying Li, and Jingyi Yu. Non-line-of-sight imaging via neural transient fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(7):2257–2268, 2021.

[35] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427–437, 2009.

[36] Bolivar Solarte, Chin-Hsuan Wu, Jin-Cheng Jhang, Jonathan Lee, Yi-Hsuan Tsai, and Min Sun. Self-training room layout estimation via geometry-aware ray-casting. In *European Conference on Computer Vision*, pages 253–269. Springer, 2024.

[37] Kun Su, Mingfei Chen, and Eli Shlizerman. Inras: Implicit neural representation for audio scenes. *Advances in Neural Information Processing Systems*, 35:8144–8158, 2022.

[38] Kushagra Tiwary, Akshat Dave, Nikhil Behari, Tzofi Klinghoffer, Ashok Veeraraghavan, and Ramesh Raskar. Orca: Glossy objects as radiance-field cameras. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20773–20782, 2023.

[39] Marco Toschi, Riccardo De Matteo, Riccardo Spezialetti, Daniele De Gregorio, Luigi Di Stefano, and Samuele Salti. Relight my nerf: A dataset for novel view synthesis and relighting of real world objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20762–20772, 2023.

[40] David Tse and Pramod Viswanath. *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[41] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. *CVPR*, 2022.

[42] Dor Verbin, Pratul P Srinivasan, Peter Hedman, Ben Mildenhall, Benjamin Attal, Richard Szeliski, and Jonathan T Barron. Nerf-casting: Improved view-dependent appearance with consistent reflections. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–10, 2024.

[43] Zian Wang, Tianchang Shen, Jun Gao, Shengyu Huang, Jacob Munkberg, Jon Hasselgren, Zan Gojcic, Wenzheng Chen, and Sanja Fidler. Neural fields meet explicit geometric representation for inverse rendering of urban scenes, 2023.

[44] Zhiwen Yan, Chen Li, and Gim Hee Lee. Nerf-ds: Neural radiance fields for dynamic specular objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8285–8295, 2023.

[45] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4578–4587, 2021.

[46] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics (ToG)*, 40(6):1–18, 2021.

[47] Xiaopeng Zhao, Zhenlin An, Qingrui Pan, and Lei Yang. Nerf2: Neural radio-frequency radiance fields. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*, pages 1–15, 2023.

[48] Chuhang Zou, Alex Colburn, Qi Shan, and Derek Hoiem. LayoutNet: Reconstructing the 3D Room Layout from a Single RGB Image . In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2051–2059, Los Alamitos, CA, USA, June 2018. IEEE Computer Society.

 https://doi.org/10.52202/085713-3743

Appendix for
Can NeRFs "See" without Cameras?

A Background on Channel Impulse Response

When a wireless signal propagates, it is typically influenced by multipath effects such as reflections and scattering, as well as attenuation caused by the surrounding environment—collectively referred to as the channel. The overall impact of these phenomena on the signal is characterized by a linear model known as the Channel Impulse Response (CIR) [28].

Mathematically, the CIR is expressed as a sum of scaled and delayed impulses as shown in Eqn 9.

$$h(t) = \sum_{i=1}^{N} a_i e^{j\phi_i} \delta(t - \tau_i), \tag{9}$$

where N is the number of multipath components, a_i denotes the amplitude (attenuation factor) of the i-th path, ϕ_i represents the phase shift of the i-th path, and τ_i is the delay of the i-th path.

For an input signal $x(t)$ transmitted through the channel $h(t)$, the output signal measured at a *Rx*, $y(t)$ is obtained by the convolution:

$$y(t) = x(t) * h(t) + w(t), \tag{10}$$

where $w(t)$ represents zero-mean additive noise. For a simple two-path channel with a line of sight (LoS) path and one reflected path, the CIR $h(t)$ is given as

$$h(t) = a_1\delta(t) + a_2 e^{j\phi_2}\delta(t - \tau_2), \tag{11}$$

The received signal $y(t)$ would then be:

$$y(t) = x(t) * \left[a_1\delta(t) + a_2 e^{j\phi_2}\delta(t - \tau_2)\right] + w(t) \tag{12}$$

B Modelling Wideband Multipath Signal Power

This section shows how received power in multipath scenarios can be approximated as a sum of powers of LoS and all other multipaths. In frequency domain, the received signal $y(t)$ at a particular receiver can be expressed as

$$Y(f_k) = H(f_k)X(f_k) + W(f_k) \tag{13}$$

Here, $X(f_k)$ and $W(f_k)$ represent the discrete Fourier transforms of the signal $x(t)$ and the additive noise $w(t)$ at subcarrier frequency f_k with $k \in \{0, 1, ..., K\}$. The channel can be written as $H(f_k) = \sum_{l=0}^{L} a_{lk} \exp(-j2\pi f_k \tau_l)$ where $l \in \{1, 2, ..., L\}$ is an index over separate multipaths. Here, a_{lk} and τ_l represents the attenuation and phase of the l-th multipath component at subcarrier k. We assume that the channel $H(f_k)$ and the signal $X(f_k)$ are independent.

The received power at each frequency k is given by $\psi_{Y_k} = \mathbb{E}[|Y(f_k)|^2]$.

$$\begin{aligned}
\mathbb{E}[|Y(f_k)|^2] &= \mathbb{E}[|H(f_k)X(f_k) + W(f_k)|^2] \\
&= \mathbb{E}[|H(f_k)|^2 |X(f_k)|^2] + \mathbb{E}[|W(f_k)|^2] + 2\text{Re}\{\mathbb{E}[H(f_k)X(f_k)W(f_k)]\} \\
&= \mathbb{E}[|H(f_k)|^2]\,\mathbb{E}[|X(f_k)|^2] + \mathbb{E}[|W(f_k)|^2] + 2\text{Re}\{\mathbb{E}[H(f_k)X(f_k)]\,\mathbb{E}[W(f_k)]\} \\
&= \mathbb{E}[|H(f_k)|^2]\,\mathbb{E}[|X(f_k)|^2] + \mathbb{E}[|W(f_k)|^2], \because \mathbb{E}[W(f_k)] = 0 \\
&= \psi_{X_k}\,\mathbb{E}[|H(f_k)|^2] + \psi_{W_k}
\end{aligned} \tag{14}$$

where $\psi_{X_k} = \mathbb{E}[|X(f_k)|^2]$ and $\psi_{W_k} = \mathbb{E}[|W(f_k)|^2]$. Next,

$$
\begin{aligned}
\mathbb{E}[|H(f_k)|^2] &= \mathbb{E}[H(f_k)H^*(f_k)]^5 \\
&= \mathbb{E}\left[\left(\sum_{l=0}^{L} a_{lk}\exp\left(-j2\pi f_k\tau_l\right)\right)\left(\sum_{m=0}^{L} a_{mk}^*\exp\left(j2\pi f_k\tau_m\right)\right)\right] \\
&= \mathbb{E}\left[\left(\sum_{l=0}^{L}\sum_{m=0}^{L} a_{lk}a_{mk}^*\exp\left(-j2\pi f_k\left(\tau_l-\tau_m\right)\right)\right)\right] \\
&= \mathbb{E}\left[\sum_{l=0}^{L}|a_{lk}|^2 + \sum_{l=0}^{L}\sum_{\substack{m=0\\m\neq l}}^{L} a_{lk}a_{mk}^*\exp\left(-j2\pi f_k\left(\tau_l-\tau_m\right)\right)\right] \\
&= \mathbb{E}\left[\sum_{l=0}^{L}|a_{lk}|^2\right] + \sum_{l=0}^{L}\sum_{\substack{m=0\\m\neq l}}^{L}\mathbb{E}\left[a_{lk}a_{mk}^*\right]\exp\left(-j2\pi f_k\left(\tau_l-\tau_m\right)\right) \\
&= \mathbb{E}\left[\sum_{l=0}^{L}|a_{lk}|^2\right]
\end{aligned}
$$

We assume that channel gains between different multipaths l, m with $l \neq m$ have vanishingly small correlation. Therefore, the total received power ψ can be computed by summing all individual subcarrier powers.

$$
\begin{aligned}
\psi &= \sum_{k=1}^{K}\psi_{Y_k} \\
&= \sum_{k=1}^{K}\psi_{X_k}\,\mathbb{E}\left[\sum_{l=0}^{L}|a_{lk}|^2\right] + \psi_{W_k} \\
&= \sum_{l=0}^{L}\sum_{k=1}^{K}\mathbb{E}\left[\psi_{X_k}|a_{lk}|^2\right] + \psi_{W_k}
\end{aligned}
$$

Here, we separate out $\sum_{k=1}^{K}\mathbb{E}\left[\psi_{X_k}|a_{0k}|^2\right] = \psi_{LoS}$, to represent the LoS power over all subcarriers and $\sum_{k=1}^{K}\mathbb{E}\left[\psi_{X_k}|a_{lk}|^2\right] = \psi_{ref_l}, 1 \leq l \leq L$, to denote the power of l-th order reflections.

C Approximating Channel with First-Order Reflections

`EchoNeRF` models the total received power at the *Rx* as the combination of the LoS power `EchoNeRF_LoS`, and the contributions from all the first-order reflections. To validate the contribution of the achievable power from `EchoNeRF` when compared to the total received power ψ, we evaluate the relative contributions of these signals to the total power using the NVIDIA Sionna simulator [15]. To this end, we compute the ratios of the LoS signal ψ_{LoS}, LoS with the first order reflections $\psi_{LoS} + \psi_{ref_1}$, and LoS with the first two orders of reflections $\psi_{LoS} + \psi_{ref_1} + \psi_{ref_2}$. These are compared to the total received power ψ, which is approximated as the sum of the LoS power and the power from the first ten reflections.

Fig 9 shows path power contribution ratio from different paths in histogram. While the ψ_{LoS} power alone only accounts for approximately 70% of the total received power and is more spread out, $\psi_{LoS} + \psi_{ref_1}$ accounts to 95% of the total power, with a reduced spread. Moreover, secondary reflections ψ_{ref_2} only contribute to less than 3% of the total power. Hence, `EchoNeRF` models the first-order reflections along with the line-of-sight.

[5] We use * to denote complex conjugate

https://doi.org/10.52202/085713-3743

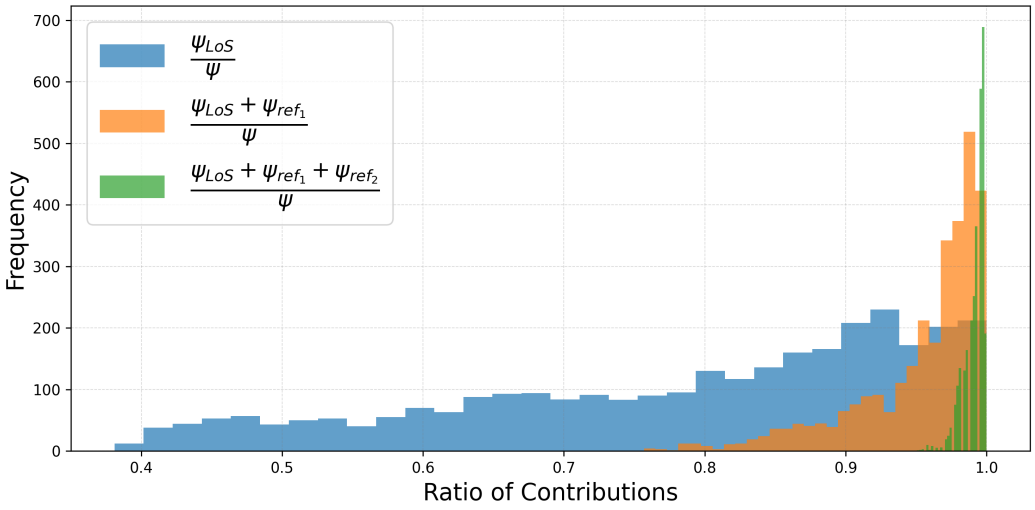

Figure 9: Histograms illustrating the contribution ratios from line-of-sight (LoS), LoS combined with first-order reflections, and LoS combined with first- and second-order reflections. The orange graph highlights the significant contribution of first-order reflections to the total power, supporting EchoNeRF's approach of modeling only the first reflection alongside the LoS power.

D Relaxing *Tx* Assumptions

We relax the assumption that *Tx* locations are known. Given the set of receiver locations $\{\mathbf{Rx}_i\}$ and the signal powers $\{\psi_i\}$, the goal is to estimate the transmitter location $\mathbf{Tx} = (\mathbf{Tx_x}, \mathbf{Tx_y})$. To achieve this, we apply a maximum likelihood estimate (MLE). Briefly, among all the measured signal powers $\{\psi_i\}$ from a given *Tx* we identify the P strongest signal powers and their corresponding received locations. The rationale behind selecting the strongest powers is that they are significantly influenced by the LoS component, allowing us to model them effectively only using the Friss' equation [1]. We assume independence among the measurements since the received LoS power across locations, for a given a *Tx* location, are independent. So, the likelihood equation for all these P measurements can be written as:

$$p(\psi_1, \psi_2, \ldots, \psi_P | \mathbf{Tx}) = \prod_{i=1}^{P} p(\psi_i | \mathbf{Tx})$$

We approximate that the ψ_i is normally distributed with a mean modeled by the line-of-sight power $\frac{K}{d_i^2}$ and variance σ^2 where $d_i = ||\mathbf{Tx} - \mathbf{Rx}_i||$ is the distance between $\mathbf{Tx}$ and $\mathbf{Rx}_i$. The likelihood function for each observation ψ_i is thus given by:

$$p(\psi_i | \mathbf{Tx}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\psi_i - \frac{K}{d_i^2})^2}{2\sigma^2}\right)$$

Maximizing log-likelihood L of $\{\psi_i\}\ \forall\ i \in \{1, \ldots, P\}$

$$\log L(\mathbf{Tx}) = \sum_{i=1}^{P} \log\left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\psi_i - \frac{K}{d_i^2})^2}{2\sigma^2}\right)\right)$$

$$\log L(\mathbf{Tx}) = -\frac{P}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{P}\left(\psi_i - \frac{K}{d_i^2}\right)^2$$

Minimizing the second term gives the optimal $\mathbf{Tx}^*$ as:

$$\mathbf{Tx}^* = \underset{\mathbf{Tx}}{\operatorname{argmin}} \sum_{i=1}^{P}\left(\psi_i - \frac{K}{\|\mathbf{Tx} - \mathbf{Rx}_i\|_2^2}\right)^2$$

We use Scipy's 'minimize' with the BFGS method to numerically solve for $\mathbf{Tx}^*$. Fig 13. visualizes the ground truth and the estimated *Tx* locations across 6 floorplans. The estimated *Tx* positions closely match the ground truth, and we report the *Tx* location error to be 2.08 pixels. Fig 10. demonstrates the performance of `EchoNeRF_LoS` and `EchoNeRF` using the estimated *Tx* locations for the 6 floorplans in Fig 13. Performance is comparable to that achieved with ground truth *Tx* locations, highlighting robustness.

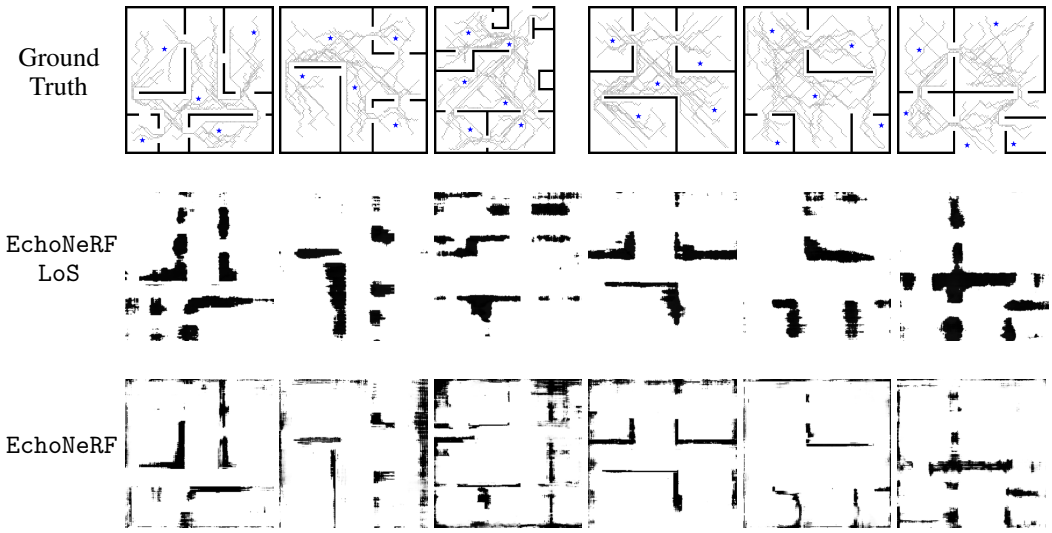

Figure 10: Qualitative comaprison of `EchoNeRF_LoS` and `EchoNeRF` when *Tx* locations are unknown, and are estimated. The top row shows the ground truth floorplan, *Rx* locations along with the estimated *Tx*s in blue. The second and third row displays the performance of `EchoNeRF_LoS` and `EchoNeRF` respectively. Despite the *Tx* locations being unknown, our methods accurately estimate them, leading to performance comparable to the case where *Tx* locations are known.

E Details on Model Training

The signal power measured at the receiver is typically represented in a logarithmic scale. RSSI values generally range from -50 dB to -120 dB, where a higher value (e.g., -50 dB) corresponds to a stronger signal. Fig 11 illustrates a typical input to `EchoNeRF` where measurements have been collected from approximately 2000 *Rx*s positioned in the floorplan, with data gathered from five *Tx*s.

E.1 Linear-Scale RSSI Loss:

For the training of `EchoNeRF`, we optimize on the linear-scale RSSI values. Linear loss ensures that the receivers that capture stronger signals are given more importance during training. We partition our dataset into an 80-20 split, using 80% of the data for model training, including baselines.

For `EchoNeRF`'s network, we employ a simple 8-layer MLP with a hidden dimension of 256 units. For each voxel v_j, the outputs from the final layer are passed through a sigmoid activation to obtain the opacity δ, and through a Gumbel softmax [16] layer to sample the output normal ω from one of the possible K_ω orientations. This sampled orientation is then used in the subsequent stages of training, such as for calculating the direction of the reflected signal M. For the learnable baselines, such as `MLP` and `NeRF2`, we adopt the same architecture as used in `EchoNeRF`.

E.2 Supervising Voxel Orientations

`EchoNeRF` leverages the spatial gradient of a voxel's opacity, $\nabla\delta$, to supervise its orientation during the multi-stage training process. To compute this gradient, we evaluate the opacities of neighboring

https://doi.org/10.52202/085713-3743

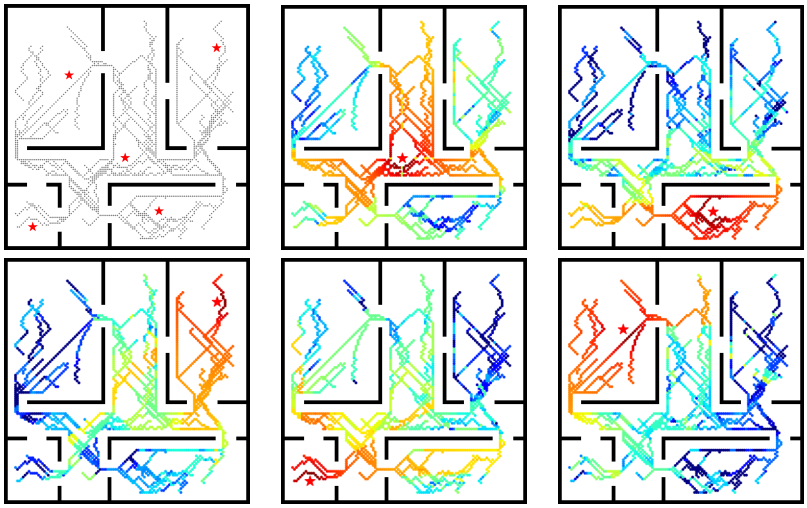

Figure 11: Observed signal power at the *Rx*s . The top left figure shows the positions of the *Rx*s and *Tx*s, followed by the power at the receivers from each of the five transmitters. The colormap ranges from red showing stronger signals to blue for weaker signals.

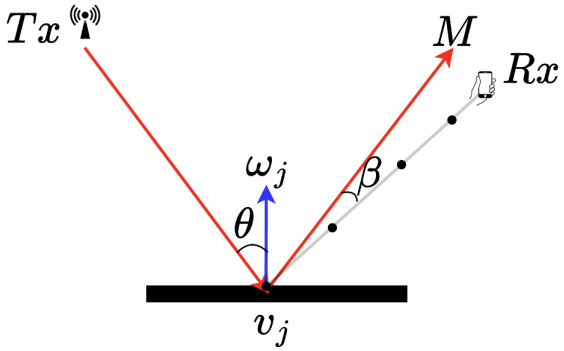

Figure 12: An incoming ray from a transmitter *Tx* reflecting around voxel v_j and arriving at receiver *Rx*. The incoming ray makes an incident angle θ with the normal ω_j to the reflecting surface. The ray after reflection passes a receiver *Rx* at a certain distance making an angle β.

voxels along each of the K_ω directions and apply finite difference methods. We found that this approach yielded superior results compared to using the gradient available via autograd.

In general, the power from reflections depends not only on the total distance traveled but also on the angle at which the reflection occurs at the voxel v_j, and whether the reflected ray reaches the receiver (*Rx*). We model this behavior through θ and β respectively which parameterize a nonlinear function f. The incidence angle θ measures the angle between the *Tx* and the orientation ω_j, and β denotes the angle *Rx* makes with the reflected ray M (see fig 12). Note that if $v_j \in \mathcal{V}$, and if ω_j is correct, $\beta = 0$. The reflected ray M can be computed as shown in Eqn 15.

$$M = (v_j - Tx) - 2\left[(v_j - Tx).\omega_j\right]\omega_j \tag{15}$$

Here ω_j is a unit vector.

E.3 Detailed network parameters

We assume the floorplan is an unknown shape inside a 512×512 grid. For any ray or ray segment, we uniformly sample $n_r = 64$ voxels on it. We choose $\lambda_1 = \lambda_2 = 0.01$ for LoS training followed by $\lambda_1 = \lambda_2 = 0.1$ for training the `EchoNeRF` model. We find that discretized opacity values to $\{0, 1\}$ improve our LoS model. We use the straight-through estimator [4] to avoid the unavailability of the gradient at the discretization step. To help optimization and to encourage sparsity of the number of reflections, we use only the top-k contributions (k=10) while training the reflection model. We use the ADAM optimizer [17] with 1.0^{-4} learning rate. We train our models on NVIDIA A100 GPUs.

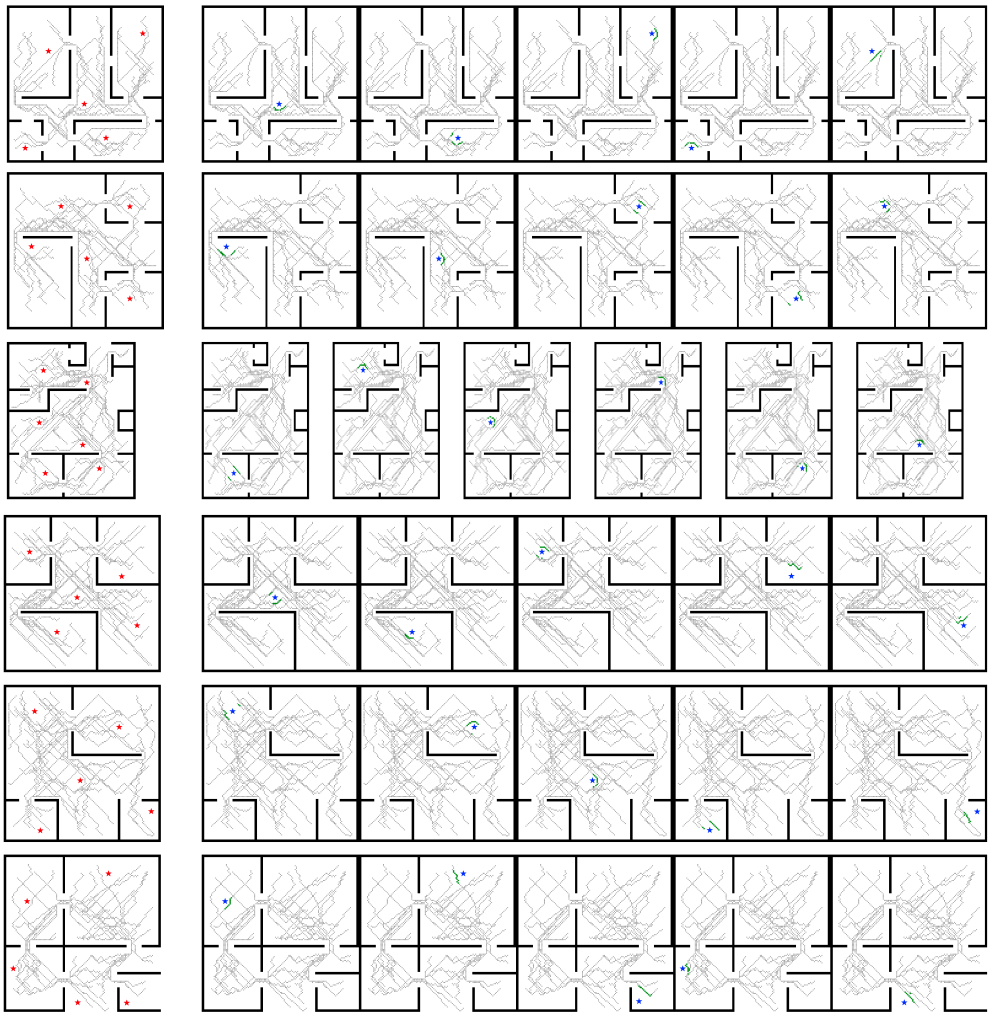

Figure 13: Comparison of Ground truth *Tx* locations indicated in red in the first column with the estimated *Tx* locations shown in blue from starting from column two. The *Rx* positions used for the estimation are marked in green.

E.4 Heatmap Segmentation Implementation Details

The raw trajectory signal power values are first interpolated to obtain a heatmap that provides a smoother representation of the input measurements. Of course, interpolating for regions without any data can lead to incorrect results, especially in larger unseen areas. A rule-based classifier is then applied for segmentation, using two criteria: (1) RSSI values above a threshold to identify potential room areas, and (2) smoothness of the RSSI signal, assessed through the second-order derivative, to ensure continuity within rooms. The initial segmentation is refined using morphological operations (via dilation and erosion) with a 3x3 kernel to smooth rough edges and eliminate small components. Overlapping regions are resolved by comparing gradient magnitudes, followed by additional morphological processing and connected component analysis to obtain the final, refined segmentation.

Code: We plan to release our code, data, and baselines soon. In the meantime, Section E provides sufficient details to allow readers to reproduce our results, especially since our network components are simple MLPs.

https://doi.org/10.52202/085713-3743

F Evaluation on Additional Floorplans

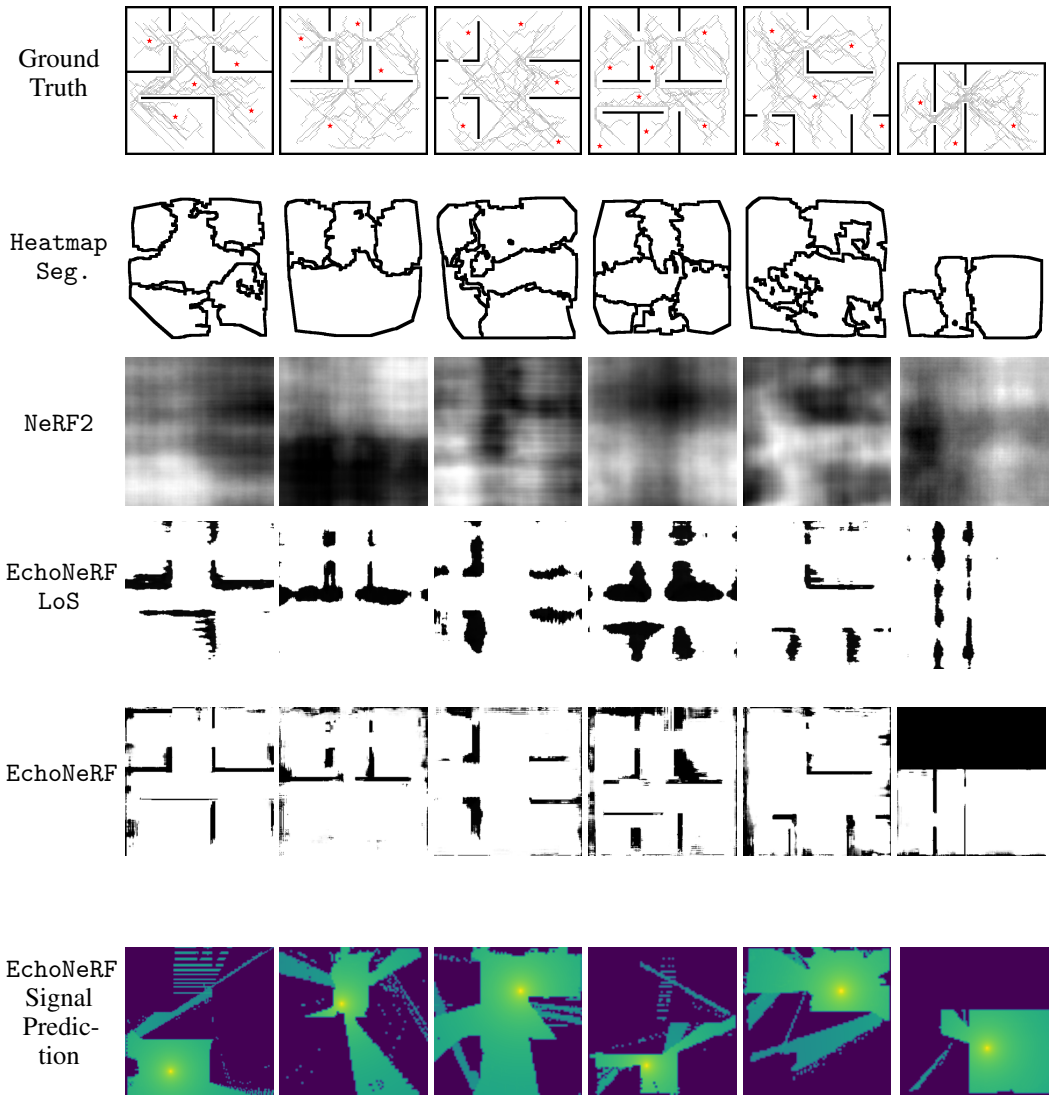

Figure 14: Qualitative comparison of Ground Truth floorplans against those inferred by baselines `Heatmap Segmentation` and `NeRF2`. We note that while `NeRF2` is unable to predcit any reasonable floorplan, `Heatmap Segmentation`́s shape is limited by the (convex hull of the) trajectory data (see bottom left of the second row, first column). Additionally, it fails to capture critical details, such as door openings. The 4th and 5th rows show floorplans by our proposed models `EchoNeRF_LoS` and `EchoNeRF`. `EchoNeRF_LoS` captures the rough shape of the floorplan, especially the interior walls, while `EchoNeRF` further improves these walls by adjusting their thickness and accurately correcting their shape. `EchoNeRF` also correctly identifies the floorplan boundary, as evidenced in the last column, where the exact boundary is captured just from the reflections. To understand the signal propagation captured by `EchoNeRF`, we place one *Tx* in each floor plan randomly (that is not present in the training data) and evaluate the signal power at discrete receivers. These *Rx*s are placed on a 2D grid at equal intervals and the predicted signal power is converted into a heatmap. The bottom row shows these inferred signal power heatmaps with the brightest point indicating the Tx location (as the *Rx* closest to the *Tx* receives the highest power). `EchoNeRF` is not only able to predict the signals well across the floorplan, but also capture the propagation paths i.e., LoS signal and the first-order reflections. For instance, in the first column, the left portion of the center hall receives power only due to the wall reflection from the left wall.

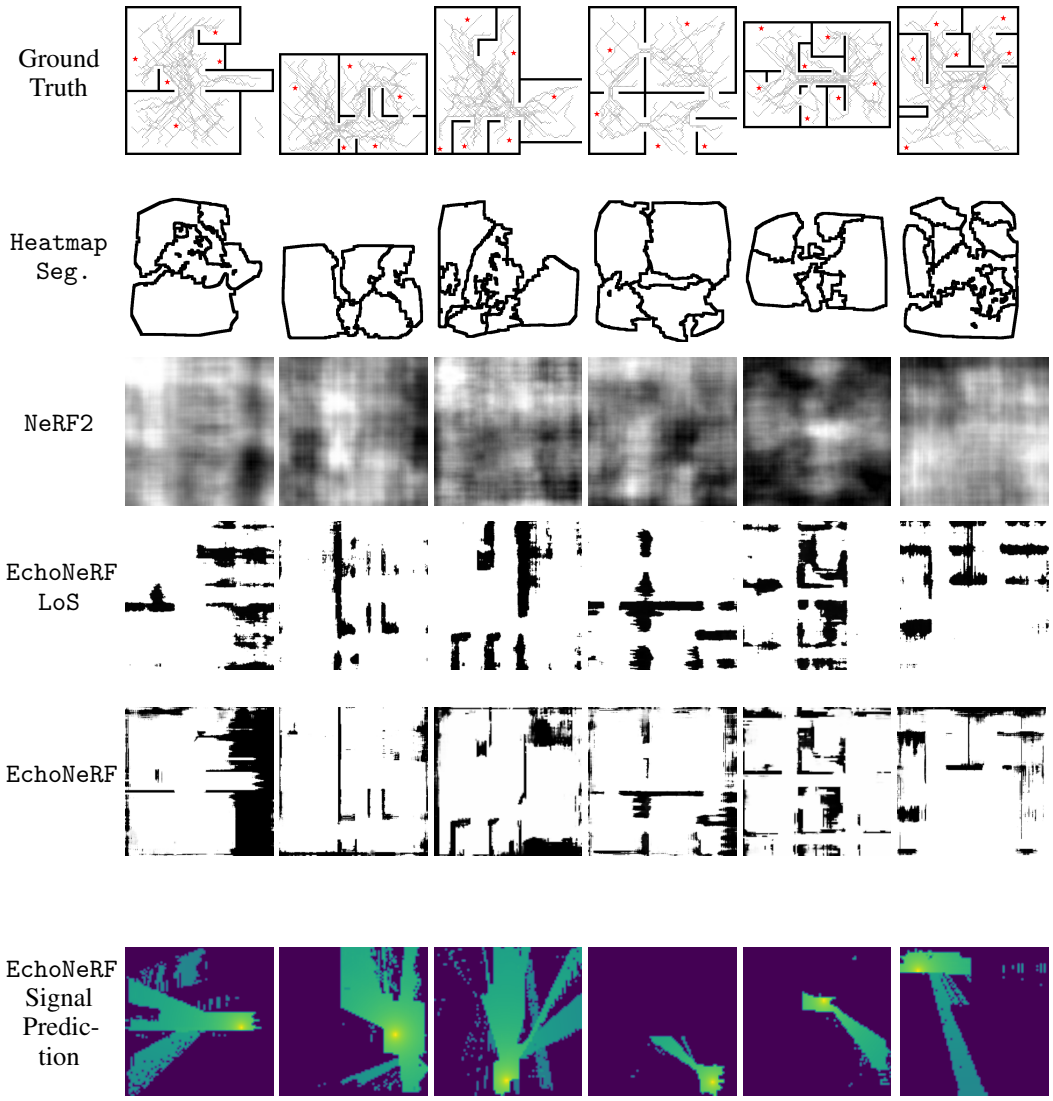

Figure 15: Additional qualitative comparisons of Ground Truth floorplans against those inferred by baselines `Heatmap Segmentation` and `NeRF2`. The 4th and 5th rows show floorplans by our proposed models `EchoNeRF_LoS` and `EchoNeRF` with clearly identified walls and boundaries. The bottom row shows inferred signal power heatmaps demonstrating `EchoNeRF`'s capability to learn accurate signal propagation.

G Societal Impact Statement

We acknowledge that NeRFs hold significant potential for positive societal impact. Applications span AR/VR, medical imaging, airport security, and education, where accurate 3D reconstructions can greatly enhance functionality and understanding. However, our work on EchoNeRF also introduces potential risks. In particular, the ability to infer detailed spatial layouts from limited sensory input could be misused to access private or sensitive floorplan information. We emphasize the importance of responsible use.

https://doi.org/10.52202/085713-3743

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 accurately describe the proposed framework `EchoNeRF` in teaching NeRFs to learn wireless reflections.

   Guidelines:

   - The answer NA 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 NA 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: Limitations are discussed in Section 5.

   Guidelines:

   - The answer NA 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: [NA]

Justification: The paper does not contain any formal proofs and theorems. The necessary derivations required for our two-stage training are shown in Section 3.3.

Guidelines:

- The answer NA 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 necessary information required to reproduce the results is provided in the Appendix Section E.

   Guidelines:

   - The answer NA 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**

https://doi.org/10.52202/085713-3743

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: [No]

Justification: We plan to release our code soon. We will also release the datasets, simulator setup, along with the baselines we have implemented. In the meantime, Section E provides sufficient detail to allow readers to reproduce our results, especially since our network components are simple MLPs.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.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://nips.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, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The required information is presented in the Appendix Section E.3, Section E.

   Guidelines:

   - The answer NA 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: Mean and variance are shown in Table 1 and robustness to noise in receiver locations is checked in Table 2.

   Guidelines:

   - The answer NA 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: Necessary details are mentioned in the Appendix Section E

   Guidelines:

   - The answer NA 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: Yes.

   Guidelines:

   - The answer NA 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: Included in Appendix G

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

https://doi.org/10.52202/085713-3743

- 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., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: NA

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All the assets used for our are cited in the paper.

    Guidelines:

    - The answer NA 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:

    Guidelines:

    - The answer NA 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: [NA]

    Justification: NA

    Guidelines:

    - The answer NA 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: [NA]

    Justification: NA

    Guidelines:

    - The answer NA 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.

https://doi.org/10.52202/085713-3743

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 rigorousness, or originality of the research, declaration is not required.

Answer: [No]

Justification: LLMs are not used for this work.

Guidelines:

- The answer NA 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 (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.