Supplementary Material for mmIR: Frequency-Space Inverse Rendering for 3D Millimeter-Wave Radar ADC Synthesis

Adnan Armouti Yixuan Gao Rajalakshmi Nandakumar
Cornell Tech, New York, NY, USA

Table of Contents

A TDM MIMO FMCW Theory . . . 1
A.1 FMCW Formulation . . . 2
A.2 IQ Demodulation for Complex Baseband ADC . . . 2
A.3 TDM MIMO for 2D and 3D Imaging . . . 4
A.4 Bistatic Radar Equation: From RCS to MC Estimator . . . 6
A.5 From Received Power to ADC Counts . . . 7
A.6 Range and Angular Resolution . . . 8
A.7 Polar-to-Cartesian Conversion . . . 10
B Implementation Details . . . 11
B.1 Input Data Format and Pre-Processing . . . 11
B.2 LiDAR-Radar Coarse-to-Fine Alignment . . . 11
B.3 Amplitude-Only Optimization and Gradient Stabilization . . . 12
B.4 Training Details . . . 13
B.5 Inference Details . . . 15
B.6 Post-Processing Details . . . 16
C Additional Results . . . 17
C.1 Material Visualization . . . 17
C.2 Surface Normal Visualization . . . 17
C.3 Antenna Beam Pattern Visualization . . . 18
C.4 Ablation Studies . . . 19
C.5 Runtime . . . 20
D Limitations . . . 21
D.1 Refraction, Absorption, and Sub-surface Scattering . . . 21
D.2 Doppler Phase Term for Moving Radar . . . 22
D.3 Dynamic Scenes and Objects . . . 22
D.4 Multi-bounce Diffraction . . . 22
D.5 Amplitude-Only Gradient Signal . . . 22
D.6 Mesh Quality Bottleneck . . . 23
D.7 Sensitivity to Sensor Alignment . . . 23

A TDM MIMO FMCW Theory

This section provides the signal-processing foundations that underpin our differentiable renderer, included for readers whose background is primarily in rendering or vision rather than radar. Sec. A.1 derives the FMCW beat signal that the renderer must synthesize per traced path. Sec. A.2 explains IQ demodulation, which

determines why the renderer outputs *complex*-valued ADC samples. Sec. A.3 shows how these ADC samples are transformed into the 2D range–azimuth and 3D range–azimuth–elevation images used in our training loss (Eq. 8) and evaluation metrics. Sec. A.4–A.5 derive the Monte Carlo estimator (main paper Eq. 3) and the power-to-ADC conversion that closes the forward model. Readers familiar with FMCW radar may skip directly to Sec. B.

A.1 FMCW Formulation

We summarize the standard FMCW signal model following [5,12]. FMCW radars emit a linearly frequency-modulated chirp into the scene,

$$s(t) = A_s \cos\!\big(2\pi f_c t + \pi K t^2\big)\,, \tag{1}$$

for $0 < t < T_c$, and receive a delayed echo

$$u(t) = A_r \cos\!\big(2\pi f_c (t-\tau) + \pi K (t-\tau)^2\big)\,, \tag{2}$$

for $\tau < t < T_c$, where K is the chirp slope, f_c is the start frequency, T_c is the chirp duration, and $\tau = 2d(t)/c$ is the round-trip delay for a target at range d and wave propagation speed c. The total swept bandwidth is

$$B = f_{\max} - f_{\min} = (f_c + KT_c) - f_c = KT_c. \tag{3}$$

When the received chirp is mixed with the (still transmitting) chirp and low-pass filtered, the resulting real-valued beat signal is

$$m_{\mathrm{R}}(t) \approx A_m \cos\!\big(2\pi K\tau\, t + 2\pi f_c \tau - \pi K \tau^2\big)\,, \tag{4}$$

for $\tau < t < T_c$, where A_m subsumes constant amplitude factors. The beat frequency is

$$f_{\mathrm{IF}} = K\tau = \frac{2Kd}{c}, \tag{5}$$

and the range-dependent phase is dominated by

$$\phi(t) \approx 2\pi f_c \tau = 4\pi \frac{d}{\lambda}, \tag{6}$$

with carrier wavelength $\lambda = c/f_c$. The small quadratic term $\pi K\tau^2$ is typically negligible for automotive-range targets ($K\tau^2 \ll f_c\tau$).

A.2 IQ Demodulation for Complex Baseband ADC

In practice, the RF front-end uses IQ demodulation as shown in Fig. 1. A single real mixer would fold the symmetric RF sidebands at $\pm f_{\mathrm{IF}}$ onto the same baseband frequencies, effectively aliasing positive and negative beat frequencies onto the same baseband bins. IQ mixing instead produces a complex-valued baseband signal with a one-sided spectrum, keeping positive and negative frequencies

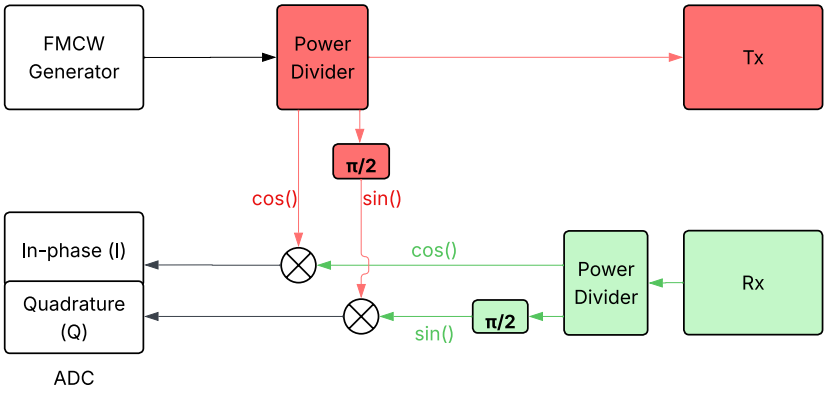


Fig. 1: Block diagram of IQ demodulation in FMCW radar. The received echo is mixed with in-phase (I) and quadrature (Q) copies of the transmit chirp and low-pass filtered, producing complex baseband samples that preserve both amplitude and phase for downstream range and angle processing.

separated, improving SNR, and yielding an unambiguous mapping between beat frequency and range [1–3].

A copy of the transmit chirp is routed to the local-oscillator (LO) ports of two mixers: one LO branch uses a cosine (in-phase, I) and the other a 90°-shifted sine (quadrature, Q). The received RF signal is applied to the RF port of both mixers, and each output is low-pass filtered and digitized, yielding

$$I(t) \propto \cos\big(2\pi f_{\mathrm{IF}} t + \phi(t)\big), \tag{7}$$

$$Q(t) \propto \sin\big(2\pi f_{\mathrm{IF}} t + \phi(t)\big). \tag{8}$$

Interpreting these as the real and imaginary parts of a single complex baseband beat signal,

$$m(t) = I(t) + j\,Q(t), \tag{9}$$

is equivalent to multiplying the received RF with a complex LO and discarding the image band.

To make this analytic form explicit, we write the transmitted and received chirps using Euler's formula following [3]:

$$s(t) = A_s \exp\big(j(2\pi f_c t + \pi K t^2)\big), \tag{10}$$

$$u(t) = A_r \exp\big(j[2\pi f_c (t-\tau) + \pi K (t-\tau)^2]\big). \tag{11}$$

Ideal IQ demodulation corresponds to mixing the received signal with the complex conjugate of the transmit signal:

$$m(t) = u(t)\, s^*(t). \tag{12}$$

Using the phase difference between $u(t)$ and $s(t)$, this simplifies to

$$m(t) = A_s A_r \exp\left(j\left[2\pi K\tau\, t + 2\pi f_c \tau - \pi K \tau^2\right]\right) \tag{13}$$
$$\approx A_m \exp\left(j\left[2\pi f_{\mathrm{IF}} t + \phi(t)\right]\right), \tag{14}$$

where again $f_{\mathrm{IF}} = K\tau = 2Kd/c$ and $\phi(t) \approx 4\pi d/\lambda$.

Finally, the ADC samples this complex baseband signal at rate f_s, producing discrete-time IQ data

$$m[n] \approx A_m \exp\left(j\left[2\pi f_{\mathrm{IF}} \frac{n}{f_s} + \phi[n]\right]\right), \tag{15}$$

where n indexes fast-time samples within a chirp and $\phi[n]$ encodes the path-length-dependent phase (and any slow motion) at the n-th sample. These complex ADC samples are the starting point for our subsequent MIMO imaging formulation.

A.3 TDM MIMO for 2D and 3D Imaging

The complex ADC samples in (15) describe a single transmit–receive pair observing a single target. Modern mmWave sensors instead use multiple transmit (Tx) and receive (Rx) antennas arranged in an array and operate them in time-division multiplexing (TDM) mode to synthesize a larger *virtual* aperture. This enables 2D and 3D spatial imaging via discrete Fourier transforms over carefully chosen dimensions of the ADC data cube, as described by [9].

TDM MIMO data cube. Let N_{Tx} and N_{Rx} denote the number of physical transmitters and receivers, N_{chirp} the number of chirps in a frame, and N_{s} the number of fast-time ADC samples per chirp. In a TDM sequence, only one Tx is active at a time. Each chirp slot is assigned to a particular transmitter, cycling through all N_{Tx} elements.

For Tx element $i \in \{0, \ldots, N_{\mathrm{Tx}} - 1\}$, Rx element $j \in \{0, \ldots, N_{\mathrm{Rx}} - 1\}$, chirp index $\ell \in \{0, \ldots, N_{\mathrm{chirp}} - 1\}$, and fast-time sample index $n \in \{0, \ldots, N_{\mathrm{s}} - 1\}$, the sampled complex beat signal can be written as

$$m_{ij}[\ell, n] \approx A_{ij}[\ell]\, \exp\left(j\left[2\pi f_{\mathrm{IF}} \frac{n}{f_s} + \phi_{ij}[\ell]\right]\right), \tag{16}$$

where f_s is the ADC sampling rate, $A_{ij}[\ell]$ captures path loss and BSDF/antenna gains, and the phase $\phi_{ij}[\ell]$ contains the range phase $4\pi d/c$ plus offsets from Tx/Rx array geometry and slow motion across chirps.

Stacking all (i, j, ℓ, n) produces a four-dimensional data block. For convenience, the Tx/Rx indices are often flattened into a *virtual* antenna index $v \in \{0, \ldots, N_{\mathrm{V}} - 1\}$ with $N_{\mathrm{V}} = N_{\mathrm{Tx}} N_{\mathrm{Rx}}$, giving $m_v[\ell, n]$.

Range FFT (fast-time). The first processing step is a 1D FFT along fast time n for each virtual channel v and chirp ℓ, which converts beat frequency to range. Before the FFT, a Hann window $w_{\mathrm{R}}[n]$ is applied in fast time to suppress spectral leakage and sidelobes:

$$M_v[\ell, k] = \sum_{n=0}^{N_{\mathrm{s}}-1} w_{\mathrm{R}}[n]\, m_v[\ell, n]\, \exp\!\left(-j\frac{2\pi}{N_{\mathrm{s}}} kn\right), \tag{17}$$

$k = 0, \ldots, N_{\mathrm{s}} - 1$. Each frequency bin k corresponds to a range

$$R_k = \frac{c\,k}{2B}, \tag{18}$$

so the magnitude $|M_v[\ell, k]|$ encodes the reflectivity at range R_k observed by virtual channel v. Range estimation is therefore driven primarily by the *amplitude* of the fast-time FFT, while the complex phase of $M_v[\ell, k]$ is carried forward for angle processing.

Angle-of-arrival from virtual array phase. At a fixed range bin k, a far-field target arriving from direction (θ, φ) (azimuth and elevation) produces nearly identical amplitudes across the virtual aperture but a systematic phase ramp tied to the projection of the direction vector onto each element's position. For a 1D linear virtual array along x with inter-element spacing d_x, the phase at element v satisfies

$$\phi_v(k) \approx \phi_0(k) + \frac{2\pi}{\lambda}\, x_v \sin\theta, \tag{19}$$

where x_v is the x-coordinate of virtual element v. Taking a DFT over the virtual array index therefore steers beams in angle:

$$\begin{aligned}\widetilde{M}[k, n_\theta] = \sum_{v=0}^{N_{\mathrm{V}}-1} w_{\mathrm{AZ}}[v] \left(\frac{1}{N_{\mathrm{chirp}}} \sum_{\ell=0}^{N_{\mathrm{chirp}}-1} M_v[\ell, k]\right) \\ \times \exp\!\left(-j\frac{2\pi}{N_{\mathrm{V}}} n_\theta v\right),\end{aligned} \tag{20}$$

where $w_{\mathrm{AZ}}[v]$ is a spatial (e.g., Hann) window applied across the virtual aperture to reduce angular sidelobes. The bin index n_θ maps to azimuth angle via the usual array steering relation,

$$\theta_{n_\theta} \approx \sin^{-1}\!\left(\frac{\lambda}{N_{\mathrm{V}} d_x}\, n_\theta\right), \tag{21}$$

assuming half-wavelength spacing and small-angle approximation. Here, the *relative phase* of $M_v[\ell, k]$ across v is what encodes direction-of-arrival. The FFT over v converts this phase gradient into an angular spectrum. The result $|\widetilde{M}[k, n_\theta]|$ is a 2D range–azimuth (RA) map.

When the physical Tx and Rx antennas are arranged in a 2D grid, TDM MIMO produces a 2D virtual array indexed by (v_x, v_z) along horizontal and

vertical baselines with spacings d_x and d_z. After the range FFT, two consecutive spatial FFTs over v_x and v_z yield a 3D range–azimuth–elevation (RAE) cube:

$$\widetilde{M}_x[k, n_\theta, v_z] = \sum_{v_x} w_{\mathrm{AZ}}[v_x]\, \bar{M}[k, v_x, v_z] \times \exp\!\left(-j\frac{2\pi}{N_x} n_\theta v_x\right), \tag{22}$$

$$\widetilde{M}_{xz}[k, n_\theta, n_\phi] = \sum_{v_z} w_{\mathrm{EL}}[v_z]\, \widetilde{M}_x[k, n_\theta, v_z] \times \exp\!\left(-j\frac{2\pi}{N_z} n_\phi v_z\right). \tag{23}$$

with N_x and N_z the number of virtual elements in azimuth and elevation, and $w_{\mathrm{EL}}[v_z]$ an elevation window. The indices (n_θ, n_ϕ) map to azimuth and elevation angles via the corresponding steering relations. In our pipeline, these RA/RAE tensors are obtained by applying Hann windows in each FFT dimension to suppress sidelobes and then using the magnitudes $|\widetilde{M}|$ for downstream 2D/3D imaging.

Slow-time (Doppler) dimension. For moving targets, the chirp index ℓ carries an additional phase term proportional to radial velocity. A further FFT along ℓ recovers Doppler frequency and yields range-Doppler-angle cubes commonly used in automotive tracking. In this work, we do not model the motion of the radar via an additional Doppler phase term, nor do we apply a Doppler FFT over all chirps. Instead, our inverse renderer assumes static scenes even though the data was collected from a moving platform, and we therefore partially attribute residual differences between rendered and ground-truth ADC to this simplification. For 2D/3D spatial imaging, we apply only the fast-time FFT for range and the virtual-array FFTs for azimuth and elevation, all acting on the same complex baseband ADC samples in (16).

A.4 Bistatic Radar Equation: From RCS to MC Estimator

This section provides the full derivation of the Monte Carlo estimator used in the main paper (Eq. 3). We follow an Rx-centric formulation where rays are traced from the receiver.

Area-domain form. For a single Tx–Rx pair observing a surface point $\mathbf{x}$ with normal $\mathbf{n}$, distances $d_t = \|\mathbf{x} - \mathbf{x}_{tx}\|$ and $d_r = \|\mathbf{x}_{rx} - \mathbf{x}\|$, and unit directions $\boldsymbol{\omega}_t$ (toward Tx) and $\boldsymbol{\omega}_r$ (toward Rx), the differential received power is:

$$P_r(\mathbf{x}) = V \cdot P_t \, \frac{G_t(\boldsymbol{\omega}_t)\, G_r(\boldsymbol{\omega}_r)\, \lambda^2}{(4\pi)^2\, d_t^2\, d_r^2} \, f(\boldsymbol{\omega}_t, \mathbf{x}, \boldsymbol{\omega}_r)\, c_t\, c_r\, dA_x, \tag{24}$$

where $V \in \{0, 1\}$ is visibility, G_t, G_r are directional antenna gains, f is the surface BSDF, and $c_t = \max(0, \mathbf{n} \cdot \boldsymbol{\omega}_t)$, $c_r = \max(0, \mathbf{n} \cdot \boldsymbol{\omega}_r)$ are cosine factors. This relates to the standard bistatic radar equation (main paper Eq. 1) by replacing the point-target RCS σ with a distributed BSDF: $\sigma = 4\pi \int f \, c_t \, c_r \, dA$.

Change of variables to receiver solid angle. When sampling directions at the receiver, we express the surface integral in receiver solid angle $d\omega_r$ via:

$$d\omega_r = \frac{c_r}{d_r^2} \, dA_x \qquad \Longleftrightarrow \qquad c_r \, dA_x = d_r^2 \, d\omega_r. \tag{25}$$

Substituting into (24), the factor $c_r \, dA_x / d_r^2$ becomes $d\omega_r$, so both d_r^2 and c_r are absorbed:

$$P_r = \frac{P_t \, \lambda^2}{(4\pi)^2} \int_{\Omega_{rx}} \frac{G_t(\boldsymbol{\omega}_t) \, G_r(\boldsymbol{\omega}_r) \, f(\boldsymbol{\omega}_t, \mathbf{x}, \boldsymbol{\omega}_r) \, c_t \, V}{d_t^2} \, d\omega_r. \tag{26}$$

Single-bounce MC estimator. Given N samples $\boldsymbol{\omega}_{r,k} \sim \rho_{rx}(\boldsymbol{\omega})$, the unbiased estimator for (26) is:

$$\hat{P}_r = \frac{P_t \, \lambda^2}{(4\pi)^2} \, \frac{1}{N} \sum_{k=1}^{N} \frac{G_t(\boldsymbol{\omega}_{t,k}) \, G_r(\boldsymbol{\omega}_{r,k}) \, c_{t,k} \, f_k \, V_k}{d_{t,k}^2 \, \rho_{rx}(\boldsymbol{\omega}_{r,k})}. \tag{27}$$

Multi-bounce generalization. For a path $x_{rx} \to x_1 \to x_2 \to \cdots \to x_D \to x_{tx}$ of depth D, we sample the initial direction at the receiver with PDF ρ_{rx}, each intermediate direction with PDF ρ_ℓ, and connect deterministically to the transmitter from x_D. Each surface-to-surface segment contributes a geometric coupling factor $\mathcal{G}_\ell = c_{\ell \to \ell+1} \, c_{\ell+1 \to \ell} / d_{\ell,\ell+1}^2$. The unbiased D-bounce estimator is:

$$\hat{P}_r^{(D)} = \frac{P_t \, \lambda^2}{(4\pi)^2} \, \frac{1}{N} \sum_{k=1}^{N} \left[\frac{G_r(\boldsymbol{\omega}_{r,k})}{\rho_{rx}(\boldsymbol{\omega}_{r,k})} \prod_{\ell=1}^{D-1} \frac{f_\ell \, \mathcal{G}_\ell}{\rho_\ell} \cdot \frac{G_t(\boldsymbol{\omega}_{t,k}) \, c_{D \to tx} \, f_D}{d_{D,tx}^2} \cdot V_k \right]. \tag{28}$$

The single-bounce case ($D = 1$) reduces to (27) when the product is empty. We use Russian roulette for unbiased path termination at configurable depth.

A.5 From Received Power to ADC Counts

The MC estimator in (27)–(28) yields the received power $\hat{P}_r$ in Watts for each path. In the main paper (Eq. 5), this is written as the complex amplitude $a_{ij}^{(p)}$ that "aggregates MC throughput." Here we make explicit how received power is converted to ADC counts in the renderer.

E-field to voltage. In a matched Z_0-ohm system ($Z_0 = 50\,\Omega$ for standard RF), the received electric field amplitude relates to power via

$$V = \sqrt{P_r \cdot Z_0}, \tag{29}$$

where V is the baseband voltage at the ADC input.

Table 1: Radar hardware parameters used in the forward model.

Parameter	Symbol	Value
Transmit power	P_t	13 dBm
Programmable Rx gain	G_{Rx}	30 dB
Antenna/PCB loss	L_{ant}	4 dB
System impedance	Z_0	50 Ω
ADC full scale	$\mathrm{ADC}_{\mathrm{fs}}$	32,768 (16-bit signed)
dBm-to-dBFS offset	–	13 dBm

Voltage to ADC counts. The ADC digitizes the baseband voltage with a full-scale range corresponding to $2^{15} = 32{,}768$ counts for a 16-bit signed converter. The conversion is

$$s_{\mathrm{ADC}} = V \cdot \mathrm{ADC}_{\mathrm{fs}}, \tag{30}$$

where $\mathrm{ADC}_{\mathrm{fs}} = 32{,}768$. Combining (29) and (30), the end-to-end scale factor from $\sqrt{\text{Watts}}$ to ADC counts is

$$\alpha_{\mathrm{ADC}} = \sqrt{Z_0} \cdot \mathrm{ADC}_{\mathrm{fs}} = \sqrt{50} \times 32{,}768 \approx 231{,}660. \tag{31}$$

Per-path ADC amplitude. For a single path p from Tx i to surface point $\mathbf{x}$ to Rx j, the complex ADC amplitude in Eq. 5 is therefore

$$a_{ij}^{(p)} = \alpha_{\mathrm{ADC}} \cdot G'_{\mathrm{Rx}} \cdot \sqrt{L_{\mathrm{ant}} \cdot \hat{P}_{r,k}^{(p)}}, \tag{32}$$

where $\hat{P}_{r,k}^{(p)}$ is the MC power estimate for path p from (27) or (28), L_{ant} is the round-trip antenna/PCB loss (linear), and $G'_{\mathrm{Rx}} = 10^{(G_{\mathrm{Rx,dB}} - \Delta_{\mathrm{dBFS}})/20}$ is the effective programmable receiver gain after accounting for the dBm-to-dBFS offset $\Delta_{\mathrm{dBFS}} = 13\,\mathrm{dBm}$. With $G_{\mathrm{Rx,dB}} = 30\,\mathrm{dB}$ and $L_{\mathrm{ant}} = -4\,\mathrm{dB}$, the effective gains are $G'_{\mathrm{Rx}} \approx 7.08$ and $\sqrt{L_{\mathrm{ant}}} \approx 0.63$.

Hardware parameters. Table 1 lists the radar hardware constants used in the forward model, following TI AWR2243 specifications.

Note on the $(4\pi)^2$ vs. $(4\pi)^3$ factor. The standard bistatic radar equation (main paper Eq. 1) uses a $(4\pi)^3$ denominator because it includes a point-target RCS σ. In the Rx-centric solid-angle formulation of (26), the change of variables from surface area to receiver solid angle (Eq. 25) absorbs one factor of d_r^2 and the cosine c_r, effectively replacing the third 4π factor. The MC estimator therefore uses $(4\pi)^2$ in the denominator, which is consistent with the distributed-BSDF formulation.

A.6 Range and Angular Resolution

This section describes the resolution terms as they relate to hardware parameters [9], which in turn informs inverse problems in radar imaging such as

super-resolution (along range, azimuth and/or elevation) as well as 2D-to-3D lifting [10].

Range resolution and windowing. For a linear FMCW chirp with effective sweep bandwidth B, the beat-frequency spacing after a K-point range FFT corresponds to a range-bin spacing

$$\Delta R \approx \frac{c}{2B}, \tag{33}$$

so two point targets at the same angle are resolvable in range if they are separated by more than ΔR [9].

Virtual aperture and angular resolution. After range processing, we form a virtual array by concatenating all $N_{\mathrm{Tx}} N_{\mathrm{Rx}}$ transmitter–receiver pairs. For a given range bin, the complex entry associated with virtual element (p, q) on the 2D grid (azimuth index p, elevation index q) can be modeled as

$$s_{pq} \propto \exp\Big(j\, \tfrac{2\pi}{\lambda} \big(p\, d_x \sin\theta \cos\phi + q\, d_z \sin\phi\big)\Big), \tag{34}$$

where (θ, ϕ) are azimuth and elevation, d_x and d_z are element spacings along x and z, and λ is the carrier wavelength. For a uniform linear array of N elements with spacing $d = \lambda/2$, the broadside 3 dB angular resolution scales approximately as

$$\Delta\theta \approx \frac{\lambda}{Nd} \approx \frac{2}{N} \text{ radians}, \tag{35}$$

with an analogous expression for elevation resolution $\Delta\phi$ when a vertical aperture is present [9].

TI MMWCAS for training vs. dense virtual array for inference. The TI MMWCAS RF-EVM board used for training realizes a time-division MIMO configuration with at most $N_\theta \approx 86$ virtual elements arranged predominantly along a single azimuth dimension and a very sparse sampling in elevation. Assuming $\lambda/2$ spacing, the ideal azimuth resolution at broadside is on the order of

$$\Delta\theta \approx \frac{2}{N_\theta} \approx 0.023 \text{ rad} \approx 1.3^\circ, \tag{36}$$

before windowing. Elevation resolution is minimal due to the sparse vertical baseline.

For 3D scene reconstruction at test time, we synthesize a dense 100×100 virtual aperture inspired by the Rohde & Schwarz QAR50 [11], with $\lambda/2$ spacing along both azimuth and elevation (see Fig. 3, right). In this case, $N_\theta = N_\phi = 100$, yielding near-isotropic degree-level resolution:

$$\Delta\theta \approx \Delta\phi \approx \frac{2}{100} \approx 0.02 \text{ rad} \approx 1.1^\circ \tag{37}$$

for a rectangular aperture. More importantly, the 2D virtual grid avoids the strong anisotropy of a 1D azimuth-only array: the point-spread function is approximately separable and well conditioned in both angular directions, enabling true 3D range–azimuth–elevation imaging via a 3D FFT on the ADC data cube.

A.7 Polar-to-Cartesian Conversion

After TDM MIMO processing, the radar data is expressed in spherical (range–angle) coordinates. For visualization and comparison with camera/LiDAR, we convert RA/RAE maps to Cartesian grids.

2D Range–Azimuth. The range–azimuth FFT yields a complex map $M[k, n_\theta]$. We use its magnitude

$$A[k, n_\theta] = |M[k, n_\theta]|$$

as samples of $A(R_k, \theta_{n_\theta})$, where $R_k = k\,\Delta R$ with ΔR from (18). Azimuth bins are obtained from a uniform grid in spatial frequency:

$$u_{n_\theta} = \frac{2n_\theta - N_\theta}{N_\theta}, \qquad \theta_{n_\theta} = \arcsin(u_{n_\theta}),$$

corresponding to the array FFT sampling in $\sin\theta$.

We build a Cartesian grid $(x_i, y_j) \in [-W, W] \times [0, D]$ with $D = N_R \Delta R$ and $W = D/2$, and associate each polar sample with planar coordinates

$$x_{k,n_\theta} = R_k \sin\theta_{n_\theta}, \qquad y_{k,n_\theta} = R_k \cos\theta_{n_\theta} - b_R,$$

where b_R is a small range bias. The scattered samples $(x_{k,n_\theta}, y_{k,n_\theta}, A(R_k, \theta_{n_\theta}))$ are then resampled onto the regular 2D grid (x_i, y_j) via bilinear interpolation, yielding a Cartesian RA image $A_{\mathrm{Cart}}(y_j, x_i)$.

3D RAE processing. For the 3D case, the range–azimuth–elevation FFT produces $M[k, n_\theta, n_\phi]$ with magnitudes

$$A[k, n_\theta, n_\phi] = |M[k, n_\theta, n_\phi]| = A(R_k, \theta_{n_\theta}, \phi_{n_\phi}).$$

Azimuth and elevation angles are obtained from spatial frequencies $u_\theta, u_\phi \in [-1, 1]$ via

$$\theta_{n_\theta} = \arcsin(u_{\theta,n_\theta}), \qquad \phi_{n_\phi} = \arcsin(u_{\phi,n_\phi}),$$

consistent with the virtual-aperture steering relations in Sec. A.6.

Each triple $(R_k, \theta_{n_\theta}, \phi_{n_\phi})$ is mapped to Cartesian coordinates in our radar-centric frame:

$$\begin{aligned} x_{k,n_\theta,n_\phi} &= R_k \cos\phi_{n_\phi} \sin\theta_{n_\theta}, \\ y_{k,n_\theta,n_\phi} &= R_k \cos\phi_{n_\phi} \cos\theta_{n_\theta} - b_R, \\ z_{k,n_\theta,n_\phi} &= R_k \sin\phi_{n_\phi}. \end{aligned}$$

We then resample these points onto a regular voxel grid (x_i, y_j, z_ℓ) using trilinear interpolation, producing a Cartesian volume $A_{\mathrm{Cart}}(x_i, y_j, z_\ell)$ that can be rendered as 3D heatmaps, iso-surfaces, or point clouds aligned with camera/LiDAR coordinates.

B Implementation Details

B.1 Input Data Format and Pre-Processing

Our method operates on synchronized radar and LiDAR data from the ColoRadar dataset [7]. We extract cascaded FMCW radar ADC data for the target frame, applying both phase and frequency calibration to the raw ADC samples before transposing to the format $(N_C, N_{Rx}, N_{Tx}, N_{ADC})$, where N_C is the number of chirps, N_{Rx} and N_{Tx} are the receiver and transmitter counts, and N_{ADC} is the ADC sample count. To construct the 3D scene geometry, we aggregate approximately 50 LiDAR frames centered temporally around the radar observation, transforming each frame to world coordinates using ground truth vehicle poses and cropping to the radar field-of-view. We remove exact duplicates and points coinciding with sensor positions, yielding a fused point cloud typically containing 100,000–500,000 points. Surface normals are estimated using GPU-accelerated radius search (FAISS) with a 0.1m neighborhood radius: for each point, we compute the covariance matrix of its neighbors and extract the eigenvector corresponding to the smallest eigenvalue, then orient normals toward the nearest LiDAR sensor position to ensure consistent outward-facing orientation.

Finally, we reconstruct a triangle mesh from the oriented point cloud using screened Poisson surface reconstruction [6]. We set the octree depth to 8, point weight to 1.0, and apply aggressive surface trimming with threshold 8.0 to remove spurious geometry far from the input points. This produces a clean mesh (typically 200,000–800,000 triangles) suitable for our differentiable ray tracing renderer. All processed data (calibrated ADC samples, individual LiDAR frames, aggregated point clouds with normals, and reconstructed meshes) are saved in standardized formats (NumPy arrays for radar/LiDAR data, PLY for meshes) for reproducibility. For each scene, JSON files specify the radar array geometry, waveform parameters, and calibrated extrinsics for rendering.

B.2 LiDAR-Radar Coarse-to-Fine Alignment

Due to residual calibration errors between the LiDAR and radar coordinate frames, we perform an automated coarse-to-fine extrinsic alignment procedure before training. This aligns the LiDAR-derived mesh with the radar observations by optimizing over a 4-DoF transformation: range translation Δr, azimuth translation $\Delta\theta$, and rotations around the elevation and azimuth axes (ϕ_{el}, ϕ_{az}).

Coarse grid search (GPU-accelerated). We first perform a coarse grid search over the 4D parameter space. For each candidate transformation, we project the LiDAR point cloud into radar range–azimuth (RA) coordinates using the transformed sensor pose, apply antenna pattern weighting, and accumulate intensity-weighted histograms into the RA grid. The resulting synthetic LiDAR-derived RA map is compared against the ground truth radar RA map using Pearson correlation as the objective:

$$\rho = \frac{\sum_{i,j}(L_{ij} - \bar{L})(R_{ij} - \bar{R})}{\sqrt{\sum_{i,j}(L_{ij} - \bar{L})^2}\sqrt{\sum_{i,j}(R_{ij} - \bar{R})^2}}, \tag{38}$$

where L and R are the min–max normalized LiDAR and radar RA maps. The grid search uses 5–7 steps per dimension with typical search ranges: $\Delta r \in [-3, 3]$ m, $\Delta\theta \in [-15°, 15°]$, $\phi_{\text{el}}, \phi_{\text{az}} \in [-10°, 10°]$. GPU acceleration via CuPy enables evaluation of ∼2500 grid points in under 2 seconds.

Fine optimization. Starting from the best coarse grid point, we refine the alignment using Nelder-Mead optimization (unconstrained, 200 iterations). The fine stage converges to sub-degree angular and sub-centimeter translational precision. Final metrics typically achieve correlation $\rho > 0.7$ between aligned LiDAR and radar RA maps, with the optimized extrinsics saved to a JSON configuration file for subsequent training.

B.3 Amplitude-Only Optimization and Gradient Stabilization

Each ADC sample is a coherent sum of multi-bounce phasors $w_i \cdot e^{j\phi_i}$, where w_i is the real-valued amplitude (BSDF, antenna gains, geometric factors) and $\phi_i \propto d_i/\lambda$ encodes the round-trip path length. At mmWave frequencies ($\lambda \approx 4\,\text{mm}$), a sub-millimeter change in path length induces an $\mathcal{O}(1)$ change in phase, making phase gradients with respect to geometry or pose extremely noisy: micrometer-scale parameter perturbations flip phasors between constructive and destructive interference, producing spiky, oscillatory gradients that dominate and destabilize optimization [4].

Amplitude-only gradient propagation. To avoid this instability, we *detach* the phase trigonometric terms from the automatic differentiation graph. Each path's contribution to the ADC is written as:

$$s_{\text{real}} = w \cdot \overline{\cos(\phi)}, \qquad s_{\text{imag}} = w \cdot \overline{\sin(\phi)}, \tag{39}$$

where $\overline{(\cdot)}$ denotes a stop-gradient operation. Gradients therefore flow from the loss through the amplitude w to material parameters, normals, and antenna patterns, but the gradient path through phase → delay → geometry is blocked. This design choice means our optimizer cannot directly exploit path-length information (a limitation we discuss further in Sec. D, Sec. D.5), but yields smooth, well-behaved gradients that converge reliably. In our experiments, amplitude-only gradients produce more stable convergence and slightly higher training correlation than full phase+amplitude gradients.

Amplitude-gradient challenges. Even with phase detached, amplitude gradients present challenges due to (i) Monte Carlo variance from stochastic path sampling, which introduces per-iteration gradient noise, and (ii) large dynamic range across parameter groups: material parameters (437,604 degrees of freedom) produce dense gradients with moderate magnitude, while antenna patterns and vertex normals have fewer parameters but can exhibit sporadic large gradients from individual high-energy paths. These effects motivate per-parameter-group gradient clipping to prevent any single group from dominating the Adam update.

Per-parameter group gradient clipping. For each parameter group θ_k, we compute the RMS gradient norm:

$$\|\nabla_{\theta_k}\mathcal{L}\|_{\text{RMS}} = \sqrt{\frac{1}{|\theta_k|}\sum_{i\in\theta_k}(\nabla_{\theta_k^i}\mathcal{L})^2}, \tag{40}$$

where $|\theta_k|$ is the total number of scalar parameters in group k. If this norm exceeds a predefined threshold τ_k, we rescale all gradients in the group by

$$\nabla_{\theta_k}\mathcal{L} \leftarrow \frac{\tau_k}{\|\nabla_{\theta_k}\mathcal{L}\|_{\text{RMS}}}\nabla_{\theta_k}\mathcal{L}. \tag{41}$$

This operation *only clips down* large gradients; when $\|\nabla_{\theta_k}\mathcal{L}\|_{\text{RMS}} \leq \tau_k$, no scaling is applied.

We set parameter-specific thresholds informed by empirical analysis of gradient magnitudes during training:

- $\tau_{\text{materials}} = 1.0$ for per-vertex material parameters,
- $\tau_{\text{pose\_rotation}} = \tau_{\text{pose\_translation}} = 0.5$ for radar pose parameters,
- $\tau_{\text{normals}} = 0.5$ for per-vertex normals,
- $\tau_{\text{patterns}} = 1.0$ for antenna beam patterns,
- $\tau_{\text{vertex\_positions}} = 0.1$ for vertex positions.

Combined with learning-rate warmup and per-parameter learning rates, gradient clipping improves convergence stability across all parameter groups.

B.4 Training Details

We optimize scene parameters via end-to-end automatic differentiation using our differentiable FMCW radar renderer. The training procedure optimizes per-vertex ITU physics material parameters (ε_r', ε_r'', σ_h, l_c, τ, slab thickness d), vertex normals, and antenna beam patterns while keeping vertex positions and radar pose fixed (Sec. 3.4). We train for 500 iterations using 1,500 reservoir hit samples per Rx antenna element, with Specular Manifold Sampling (SMS) for specular paths and free-space diffraction (FSD) BSDF [13] for edge diffraction.

The primary loss is min–max normalized range–azimuth L2 (Eq. 8). All $\sqrt{\cdot}$ operations in the AD pipeline use epsilon guards (10^{-20}) to prevent infinite gradients at zero-crossings.

Per-group learning rates and gradient clipping. Learning rates are set per parameter group: materials = 0.5, pose rotation = 5×10^{-3}, pose translation = 5×10^{-4}, vertex normals = 0.01, antenna patterns = 0.05, vertex positions = 5×10^{-4}. We apply per-group RMS gradient clipping (materials= 1.0, pose rotation= 0.5, pose translation= 0.5, normals= 0.5, patterns= 1.0, vertex positions= 0.1) to stabilize training against MC variance and inter-group dynamic range differences (Sec. B.3). A linear learning rate warmup (0.3× to 1× over 5 iterations) prevents initial divergence. Post-step constraints clamp pose translation to $\pm\lambda/2$ and rotation to $\pm2^\circ$ per iteration.

Table 2: Training hyperparameters used for all seven scenes. Renderer and optimizer settings are shared across scenes; only input data paths differ.

Hyperparameter	**Value**
Renderer	
Polarization	Vertical
Max bounces	3
Reservoir hits per Rx	1,500
Hemisphere sampling	Cosine
SMS (specular manifold)	Enabled (20 iter, tol 10^{-5})
Edge diffraction (FSD)	Enabled
Russian roulette	Start bounce 3, $p = 0.5$
Optimizer	
Iterations	500
Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
LR warmup	$0.3\times \rightarrow 1\times$ over 5 iterations
Material parameterization	Per-vertex, 6 ITU params
Initial material	Concrete (ITU-R P.2040)
Error-map initialization	Enabled
Pattern mode	Shared (1 Tx + 1 Rx pattern)
Optimized parameters	
Materials	✓
Vertex normals	✓
Antenna patterns	✓
Pose (rotation / translation)	Frozen
Vertex positions	Frozen
Loss	
Primary loss	Min–max normalized RA L2
Phase gradient	Detached (amplitude only)

Within the material parameter group, per-column learning rate scales further modulate the Adam update for each of the six material parameters (ε_r': 0.3, ε_r'': 0.5, σ_h: 0.3, l_c: 0.3, τ: 0.5, d: 0.1). These scales reflect the relative sensitivity of the loss to each parameter; in particular, slab thickness d receives the smallest scale (0.1) because its effect on slab Fresnel resonance is highly nonlinear.

Material parameterization. Each vertex stores 6 ITU physics parameters reparameterized for unconstrained optimization. Table 3 summarizes the transforms and bounds.

Initialization uses ITU-R P.2040 reference values for common materials: concrete ($\varepsilon_r' = 5.31$, $\varepsilon_r'' = 0.29$), glass ($\varepsilon_r' = 6.27$, $\varepsilon_r'' = 0.15$), and metal ($\varepsilon_r' = 1.0$, $\varepsilon_r'' = 10^7$) at 77 GHz. For a typical scene with 72,934 vertices, this yields 437,604 material degrees of freedom.

Table 3: Material parameter reparameterization. The optimizer operates in unconstrained space x; each parameter is mapped to a bounded physical value via the listed transform. Clamp bounds are applied before the exponential to prevent overflow.

Parameter	Transform	Physical range	Clamp
ε_r'	$1.5 + 8.5\,\sigma(x)$	$[1.5, 10]$	–
ε_r''	$e^{\mathrm{clamp}(x)}$	$[10^{-3}, 9{\times}10^{6}]$	$[-7, 16]$
σ_h (m)	$e^{\mathrm{clamp}(x)}$	$[10^{-7}, 10^{-3}]$	$[-16, -7]$
l_c (m)	$e^{\mathrm{clamp}(x)}$	$[5{\times}10^{-4}, 0.1]$	$[-7.6, -2.3]$
τ	$0.05 + 0.9\,\sigma(x)$	$[0.05, 0.95]$	–
d (m)	$e^{\mathrm{clamp}(x)}$	$[10^{-3}, 0.3]$	$[-7, -1.2]$

Error-map-driven material initialization. Before gradient-based training begins, we optionally run a single non-differentiable forward pass with the default material initialization and compare the rendered RA map against the ground truth. Triangles whose rendered intensity significantly exceeds the ground truth (over-reflecting phantom triangles) are identified via a per-triangle error map, and their initial material parameters are pre-adjusted (increasing imaginary permittivity ε_r'' and roughness σ_h) to reduce their reflected energy before optimization. This prevents the optimizer from wasting early iterations correcting large phantom contributions inherited from imperfect mesh reconstruction.

Antenna pattern mode. All Tx and Rx elements share a single learnable beam pattern ("shared" mode). The pattern is initialized from reference MMWCAS element patterns and jointly optimized with material and normal parameters during training. This reduces the pattern degrees of freedom from $N_{\mathrm{Tx}} + N_{\mathrm{Rx}}$ independent patterns to a single shared pattern, which is appropriate given the uniform element design of the TI MMWCAS board.

Training each scene requires approximately 10–15 minutes on a single NVIDIA RTX 4090 GPU (24 GB). Checkpoints are saved every 50 iterations, and the iteration with the lowest training loss is saved separately.

Amplitude-only gradient propagation. As detailed in Sec. B.3, we detach the phase trigonometric terms from the AD graph and propagate gradients only through the real-valued amplitude w of each phasor. This blocks the noisy phase→delay→geometry gradient path while preserving smooth, stable amplitude gradients to material parameters, normals, and antenna patterns.

B.5 Inference Details

Dense Array Configuration To generate dense radar data while maintaining consistency with the MMWCAS sensor board frame, we first create dense antenna array configurations using a uniform virtual antenna array generator. Starting from the original MMWCAS config (12Tx×16Rx), we generate a dense array with 100Tx×100Rx elements arranged in a two-column Tx and two-row Rx pattern as

shown in Figure 3 (right). The dense antennas are mapped to the same board plane as the original MMWCAS array using PCA-based board frame extraction to recover the exact horizontal, vertical, and boresight axes from the antenna positions. All dense antennas are positioned with half-wavelength ($\lambda/2 = 1.97$mm at 76 GHz) spacing and collectively translated so the dense array's geometric center exactly matches the original MMWCAS board center, preserving both position and orientation.

Dense Array ADC Generation For ADC generation, we employ batched rendering with 100 Tx-Rx pairs per batch to manage GPU memory when processing the 10,000-element MIMO array. We trace 200 reservoir hit samples per Rx element with reflection enabled. Since the goal at test time is 3D scene reconstruction, we follow [8] and use single-bounce propagation (max path depth = 1, i.e., one surface interaction per path) at test time with this dense radar array. The forward model uses antenna pattern-weighted importance sampling and loads all trained scene parameters (materials, normals, and learned antenna beam patterns) from optimization checkpoints. The renderer accumulates all path contributions (diffuse reflection, specular reflection via SMS, and edge diffraction via FSD) into a single coherent complex ADC tensor of shape (100 Tx, 100 Rx, 256 ADC samples, 2). Rendering a single ADC frame for the dense radar array takes approximately 6 seconds on an RTX 4090.

B.6 Post-Processing Details

Once the dense radar array ADC data is generated, we apply full FMCW processing: range FFT produces a 3D complex cube, virtual array mapping creates a $100 \times 100 \times 256$ tensor, followed by azimuth and elevation FFTs yielding a $256 \times 127 \times 127$ Range-Azimuth-Elevation (RAE) cube. We extract the board frame (azimuth, range, elevation axes) from antenna positions, which is then used to align the local RAE cube to the global 3D Cartesian coordinate frame. In summary, for the radar point cloud extraction, we: (1) compute magnitude RAE, (2) threshold at the 99.9th percentile, (3) convert to metric 3D coordinates, and (4) apply rigid transformation (translation + rotation) to align with scene ground truth.

Why the 3D occupancy evaluation is not circular. The 3D occupancy comparison involves three *distinct* artifacts. (i) The *input* is a Poisson mesh aggregated from $\sim$50 LiDAR sweeps (Sec. B.1), a smoothed geometric scaffold on which mmIR optimizes only materials, normals, and beam patterns. (ii) The *evaluation ground truth* is a single middle-index LiDAR frame (raw points), voxelized and clipped to the radar field of view; it is a different artifact at a different time instant with different topology from the input mesh, and is the fairer comparison since the dense-array render is itself a single-viewpoint capture. (iii) mmIR's *output* is not the input mesh but the 99.9^{th}-percentile RAE intensity-peak set synthesized by the dense virtual array, gated by radar physics (BSDF, beam pattern, FOV, occlusion, multipath), so only surfaces with strong radar returns appear. The



Fig. 2: Per-vertex material deviation from ITU-R P.2040 concrete defaults after optimization, shown across all seven scenes. Color encodes the normalized mean deviation across six physics parameters (ε_r, ε_i, σ_h, ℓ_c, τ, d) using the inferno colormap, where black = zero deviation (unchanged from defaults) and bright yellow = maximum deviation within each parameter's physical range. Log-scale parameters (ε_i, σ_h, ℓ_c, d) are normalized in log-space; linear parameters (ε_r, τ) in linear-space. Deviations concentrate on radar-visible surfaces and diminish in occluded or shadow regions.

comparison is therefore not circular. Cross-sensor transfer to the AWR1843 (Tab. 3: 0.554 vs. 0.114 for Sionna-RT) confirms the optimized parameters are sensor-agnostic, not memorized LiDAR.

C Additional Results

This section presents qualitative visualizations and quantitative ablations that complement the main paper's evaluation. We first visualize the learned parameters, including per-vertex materials (Sec. C.1), surface normals (Sec. C.2), and antenna beam patterns (Sec. C.3),to provide interpretability for the optimized representations. We then ablate each component of the full model (Sec. C.4) to quantify individual contributions and report runtime breakdowns (Sec. C.5).

C.1 Material Visualization

As shown in Fig. 2, material deviations are spatially correlated with radar visibility: surfaces under direct illumination show the largest updates, while occluded regions retain their ITU defaults. This pattern is consistent across all seven scenes despite varying geometry and viewpoints.

C.2 Surface Normal Visualization

Figure 3 shows that angular deviations are spatially concentrated at mesh boundaries, thin structures, and regions with irregular tessellation, all of which are known failure modes of Poisson surface reconstruction from sparse LiDAR returns. Mean angular deviation across the seven scenes is 9.97°. Figure 4 provides the corresponding world-space normal comparison: large planar regions (walls, ground) remain largely unchanged, while edges and thin features show visible reorientation. This spatial pattern indicates that normal optimization corrects tessellation artifacts rather than overfitting to measurement noise.

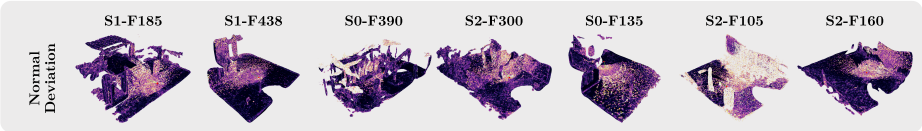


Fig. 3: Per-vertex angular deviation between initial (LiDAR-derived) and optimized surface normals across all seven scenes. Color encodes the absolute angle between initial and optimized normal vectors using the inferno colormap, where black = 0° (no change) and bright yellow = 30° or greater (clamped at 30°). Larger deviations appear primarily at mesh boundaries and on surfaces with irregular tessellation.

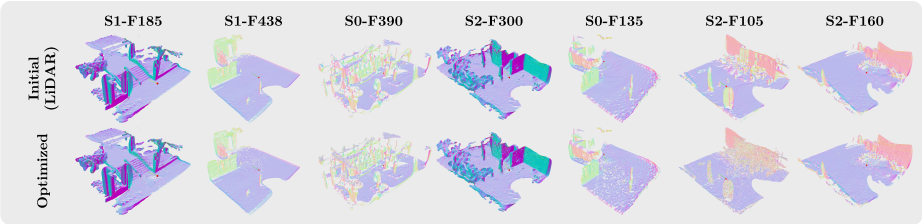


Fig. 4: World-space surface normal comparison across all seven scenes. Each vertex is colored by its unit normal direction mapped to RGB (X→R, Y→G, Z→B; value range $[-1, 1] \rightarrow [0, 1]$). Top row ("Initial"): normals computed from the Poisson-reconstructed LiDAR mesh. Bottom row ("Optimized"): normals after joint optimization. Differences between rows show where surface orientations changed during optimization; changes are most apparent at mesh boundaries and on large planar surfaces where Poisson reconstruction artifacts are common.

C.3 Antenna Beam Pattern Visualization

As shown in Fig. 5, the optimized beam patterns retain the main-lobe beamwidths of the initial reference patterns for both Tx and Rx elements across elevation and azimuth, while adjusting absolute gain levels. Per-scene variation in the optimized patterns reflects independent compensation for calibration uncertainty. The consistent preservation of main-lobe structure across all seven scenes supports the interpretation that pattern optimization primarily refines gain calibration rather than learning physically implausible beam shapes.

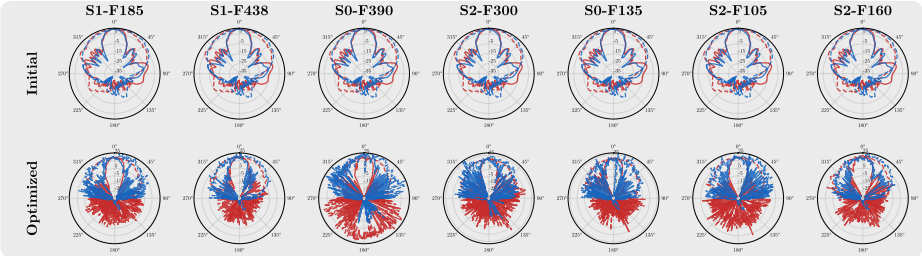


Fig. 5: Antenna beam pattern comparison across all seven scenes (polar plots, dB scale). Each plot shows four curves: Tx elevation (red solid), Tx azimuth (red dashed), Rx elevation (blue solid), and Rx azimuth (blue dashed). Top row ("Initial"): reference patterns shared across all scenes. Bottom row ("Optimized"): per-scene optimized patterns. Optimization adjusts main-lobe gain and sidelobe structure to compensate for calibration uncertainty in the reference antenna data. Per-scene variation in the optimized patterns is expected, as each scene compensates independently for calibration uncertainty. All values floored at the initial pattern minimum (-32 dB).

C.4 Ablation Studies

Table 4: Component ablation averaged over seven training scenes (500 iterations each). Each row disables one component from the full mmIR model. *Random material init* instead seeds optimization from random per-vertex materials (error-map init disabled, cf. w/o error-map init). Best per-column in **bold**.

Configuration	Corr ↑	PSNR ↑	SSIM ↑	RMSE ↓
Full model	0.919	37.6	0.902	0.014
w/o diffraction	0.910	37.4	0.896	0.014
w/o SMS	0.915	37.8	0.898	0.014
w/o multi-bounce	**0.936**	**38.7**	**0.920**	**0.012**
w/o normal optim.	0.919	37.4	0.909	0.014
w/o pattern optim.	0.885	36.6	0.882	0.015
w/o error-map init	0.909	37.4	0.895	0.014
Random material init	0.897	37.0	0.885	0.015

We ablate each component by disabling it from the full model and retraining on all seven scenes for 500 iterations (Table 4). We group ablations into three categories: rendering transport, optimization targets, and initialization.

Rendering transport. Disabling *diffraction* degrades RA correlation (from 0.919 to 0.910, -1.0%), indicating that edge-diffracted energy contributes in outdoor scenes with building edges, fences, and vehicle contours. Disabling *SMS* has a marginal effect on RA cartesian correlation metrics (from 0.919 to 0.915),

consistent with the predominantly rough surface scattering expected at 77 GHz for outdoor building materials; SMS primarily benefits scenes with smooth metallic surfaces (e.g., vehicle panels), which constitute a small fraction of the total scattering area. Restricting propagation to *single-bounce* slightly improves RA correlation (from 0.919 to 0.936) and RMSE (from 0.014 to 0.012). This suggests that higher-order bounces introduce Monte Carlo variance at the current sample budget that slightly degrades metrics, though indirect paths (e.g., ground–wall or wall–vehicle interactions) contribute measurable energy in the captured data.

Optimization targets. Disabling *antenna pattern optimization* reduces PSNR by 1.0 dB (from 37.6 to 36.6) and degrades RA correlation (from 0.919 to 0.885, −3.7%), indicating that jointly learning beam patterns compensates for calibration uncertainty in the initial reference antenna data. As shown in Fig. 5, the optimized patterns retain the main-lobe structure and beamwidths of the initial reference patterns for both Tx and Rx elements across elevation and azimuth, indicating that optimization refines gain calibration while preserving the physical beam structure. The consistent main-lobe preservation across all seven scenes is consistent with the observation that disabling pattern optimization primarily degrades PSNR (absolute power calibration) rather than correlation (spatial structure). Freezing *surface normals* at their LiDAR-derived values has negligible effect on RA correlation (from 0.919 to 0.919), while slightly improving SSIM (from 0.902 to 0.909). As visualized in Figs. 3 and 4, normal optimization concentrates adjustments at mesh boundaries and thin structures (known failure modes of Poisson surface reconstruction), with mean angular deviation of 9.97° across scenes, while leaving large planar regions largely unchanged. This spatial pattern suggests that normal optimization targets tessellation artifacts rather than overfitting to measurement noise.

Initialization. Disabling *error-map initialization* has marginal effect on final metrics (from 0.919 to 0.909), indicating that 500 gradient iterations are sufficient to largely recover from uniform material initialization. The benefit of error-map seeding is faster early convergence rather than improved final quality. As a stronger stress test, we replace the uniform material initialization with *random* per-vertex materials (error-map seeding disabled): optimization still recovers to 0.897 mean RA correlation, only 0.012 below the matched error-map-off baseline (0.909) and 0.022 below the full model (0.919). Even when a scene begins essentially uncorrelated (one starts at correlation ≈0.01), it converges to nearly the same fit, confirming that mmIR's reconstruction is driven by optimization rather than material initialization.

C.5 Runtime

Table 5 reports the runtime breakdown for our pipeline and the Sionna-RT baseline on scene S0-F135 (∼72K vertices) using a single NVIDIA RTX 4090 GPU (24 GB). Per-scene preprocessing (LiDAR aggregation, Poisson meshing,

and radar–LiDAR alignment) is a one-time cost shared by both methods. During training, our forward render averages 0.65 s per iteration, roughly 4.5× faster than Sionna-RT's 2.93 s, owing to our Taichi-based SBR kernel and cached specular-path reuse. The AD backward pass adds only 0.10 s. Including gradient clipping, the Adam optimizer step, and metric computation, the total per-iteration cost is 1.32 s, enabling 500 iterations in ∼11 min wall-clock time. At inference, rendering the full dense virtual array (10,000 elements) and computing the RAE FFT takes 6.4 s per scene.

Table 5: Runtime comparison and breakdown on a single NVIDIA RTX 4090 GPU (24 GB). Training times are averaged over 500 iterations (mmIR) and 150 iterations (Sionna-RT) on scene S0-F135 (∼72K vertices), excluding the first 5 iterations (JIT compilation warmup).

Stage	**mmIR (Ours)**	**Sionna-RT**
Per-scene preprocessing (one-time)		
LiDAR aggregation & meshing	144 s	
LiDAR–radar alignment	2.2 s	
Training (per iteration, mean ± std)		
Forward render	0.65 s	2.93 s
AD backward + grad. extraction	0.10 s	0.35 s
Grad. clipping + optimizer + metrics	0.57 s	–
Total per iteration	1.32 ± 0.02 s	3.28 ± 0.46 s
Training (total)		
Total wall-clock	11.0 min	8.2 min
Iterations	500	150
Inference (one-time per scene)		
Dense array ADC rendering	6.2 s	
RAE FFT + point cloud	0.2 s	

D Limitations

D.1 Refraction, Absorption, and Sub-surface Scattering

Currently, mmIR only models propagation in free space between surface interactions, with interfaces handled by a surface BSDF and diffraction. Transmission into volumes (including glass, plastic, drywall among others), internal attenuation, and sub-surface scattering are not simulated. In effect, any refracted component is absorbed at the first interface. This is reasonable for predominantly metallic and high-contrast scenes, but less accurate for semi-transparent or volumetric media. Supporting these effects would require explicit volumetric permittivity and internal transport via Bidirectional Scattering Distribution Functions (BSDFs), which account for both reflection and transmission.

D.2 Doppler Phase Term for Moving Radar

Our FMCW model assumes a static radar pose within each frame and attributes phase only to geometric path length. Ego-motion during a chirp burst introduces an additional Doppler phase term that we do not explicitly model. As a result, mmIR is best matched to quasi-static captures; fast platform motion will induce residual phase errors that cannot be fully absorbed by geometry or material calibration without a time-resolved pose model. Incorporating ego-motion would require a TDM-aware motion model that accounts for platform displacement between successive transmit slots.

D.3 Dynamic Scenes and Objects

We optimize a single static mesh with static material parameters to render all ADC measurements. Temporally varying objects (including vehicles, people, vibrating structures) are treated as noise and may appear as blurred or ghosted structure in 3D. To the best of our knowledge, the outdoor scenes chosen from the ColoRadar dataset for our experiments do not feature any moving objects. Accounting for dynamic scenes requires per-frame geometry, deformable models, or an explicit separation of static and dynamic components, which is not currently included and left for future work.

D.4 Multi-bounce Diffraction

Diffraction is modeled via the free-space diffraction (FSD) BSDF [13] at single-bounce surface interactions, combined with multi-bounce specular/diffuse reflections. Higher-order diffraction chains (edge–edge, edge–surface–edge) and more detailed grazing-angle behavior are not modeled. Extending to multi-bounce diffraction requires preserving Jones-vector transport along diffracted paths, which the FSD BSDF framework naturally supports. Multi-bounce diffraction may contribute additional energy in highly cluttered or strongly shadowed environments that the current single-bounce model does not capture.

D.5 Amplitude-Only Gradient Signal

As described in Sec. B.3, each ADC sample contains a phase term $\phi \propto d/\lambda$ that is extremely sensitive to path-length changes: at 77 GHz ($\lambda \approx 4\,\mathrm{mm}$), a sub-millimeter displacement induces an $\mathcal{O}(1)$ change in phase, and a half-wavelength shift produces a full 2π wrap. In our inverse-rendering setting, small parameter updates change individual path lengths by micrometers, flipping phasor contributions between constructive and destructive interference and producing a highly oscillatory, speckle-like loss landscape with narrow local minima spaced at fractions of the wavelength.

Our default training configuration therefore detaches the phase trigonometric terms $\cos(\phi)$ and $\sin(\phi)$ from the AD graph, propagating gradients only through the amplitude w of each phasor contribution (Sec. B.3). While this improves

training stability and convergence, it limits the information available to the optimizer: the loss cannot directly drive parameter updates based on path-length discrepancies. In principle, phase gradients carry complementary geometric information that amplitude alone cannot provide. Enabling full phase+amplitude gradients may benefit future work on joint geometry and material optimization, provided the phase gradient noise can be sufficiently mitigated, for instance via larger sample counts, frequency-dependent phase smoothing, or learned gradient scaling.

D.6 Mesh Quality Bottleneck

mmIR is bottlenecked by the quality of the LiDAR-derived mesh used as scene geometry. If the Poisson surface reconstruction introduces artifacts (missing surfaces, incorrect topology, or smoothed-out fine structure), the rendered ADC may diverge from the real measurement despite material or pose optimization. Thin structures (fences, poles, foliage) and regions with sparse LiDAR returns are particularly susceptible to reconstruction error, and may produce *phantom reflectors*: mesh triangles that do not correspond to any real-world surface but still contribute spurious radar returns. Our learnable energy gate $A(\boldsymbol{\omega}_t)$ (Eq. 6) partially mitigates this: optimization can drive $A \to 0$ for phantom triangles by learning high imaginary permittivity or thin-slab resonance, effectively suppressing their reflected energy.

We further employ an error-map-driven material initialization that identifies over-reflecting triangles (where rendered intensity exceeds ground truth) before gradient-based training and pre-adjusts their roughness and permittivity to reduce phantom contributions. Furthermore, LiDAR ($\lambda \sim 900$ nm) and mmWave radar ($\lambda \sim 4$ mm) interact with surfaces at fundamentally different scales: surfaces that produce strong LiDAR returns (e.g., foliage, thin fabric) may be effectively transparent at mmWave frequencies, while surfaces that are specular at mmWave (e.g., rough concrete) scatter diffusely in LiDAR. The mesh therefore includes surfaces with negligible radar cross-section and may miss structures that are radar-visible but LiDAR-dark. Our learnable energy gate partially compensates by suppressing radar-invisible surfaces, but cannot reconstruct geometry absent from the mesh. Nevertheless, these mechanisms cannot recover surfaces that are missing from the mesh entirely. Mitigating this limitation requires either higher-fidelity meshing pipelines (e.g., neural implicit surfaces, multi-frame aggregation) or jointly optimizing coarse mesh topology during inverse rendering.

D.7 Sensitivity to Sensor Alignment

The inverse rendering task is highly sensitive to the LiDAR–radar alignment, both for the cascaded imaging radar used during training and the single-chip radar used for transfer evaluation. Small translational or rotational errors shift rendered paths relative to measured ones, introducing systematic phase errors that material optimization alone cannot absorb. Our coarse-to-fine GPU alignment mitigates gross misalignment but may converge to local optima in scenes with ambiguous

geometry (flat surfaces, repetitive structure). This sensitivity is amplified for multi-chip cascade configurations where inter-chip calibration errors compound across the virtual array.

References

1. Alizadeh, M.: Remote vital signs monitoring using a mm-wave fmcw radar (2019)
2. Alizadeh, M., Shaker, G., Almeida, J.C.M.D., Morita, P.P., Safavi-Naeini, S.: Remote monitoring of human vital signs using mm-wave fmcw radar. IEEE Access **7**, 54958–54968 (2019). https://doi.org/10.1109/ACCESS.2019.2912956
3. Budge, M., Burt, M.: Range correlation effects in radars. In: The Record of the 1993 IEEE National Radar Conference. pp. 212–216 (1993). https://doi.org/10.1109/NRC.1993.270463
4. Chen, X., Ding, J., Zheng, K., Fang, X., Zhang, X., Lu, C.X., Li, Z.: Invertwin: Solving inverse problems via differentiable radio frequency digital twin (2025), https://arxiv.org/abs/2508.14204
5. Dudek, M., Wahl, R., Kissinger, D., Weigel, R., Fischer, G.: Millimeter wave fmcw radar system simulations including a 3d ray tracing channel simulator. In: 2010 Asia-Pacific Microwave Conference. pp. 1665–1668 (2010)
6. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the Fourth Eurographics Symposium on Geometry Processing. p. 61–70. SGP '06, Eurographics Association, Goslar, DEU (2006)
7. Kramer, A., Harlow, K., Williams, C., Heckman, C.: Coloradar: The direct 3d millimeter wave radar dataset. The International Journal of Robotics Research **41**(4), 351–360 (2022). https://doi.org/10.1177/02783649211068535, https://doi.org/10.1177/02783649211068535
8. Kung, P.C., Harisha, S., Vasudevan, R., Eid, A., Skinner, K.A.: Radarsplat: Radar gaussian splatting for high-fidelity data synthesis and 3d reconstruction of autonomous driving scenes. In: 2025 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 27596–27606 (2025). https://doi.org/10.1109/ICCV51701.2025.02562
9. Li, X., Wang, X., Yang, Q., Fu, S.: Signal processing for tdm mimo fmcw millimeter-wave radar sensors. IEEE Access **9**, 167959–167971 (2021). https://doi.org/10.1109/ACCESS.2021.3137387
10. Li, Y.J., Hunt, S., Park, J., O'Toole, M., Kitani, K.: Azimuth super-resolution for fmcw radar in autonomous driving. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 17504–17513 (2023). https://doi.org/10.1109/CVPR52729.2023.01679
11. Rohde & Schwarz: R&S QAR50 Quality automotive radome tester. https://www.rohde-schwarz.com/products/test-and-measurement/radome-tester/rs-qar50-quality-automotive-radome-tester_63493-1138625.html (2023), accessed: 2025-12-05
12. Skolnik, M.I., et al.: Introduction to radar systems, vol. 3. McGraw-hill New York (1980)
13. Steinberg, S., Ramamoorthi, R., Bitterli, B., Mollazainali, A., D'Eon, E., Pharr, M.: A free-space diffraction bsdf. ACM Trans. Graph. **43**(4) (Jul 2024). https://doi.org/10.1145/3658166, https://doi.org/10.1145/3658166