

Sionna RT: Technical Report

Fayçal Aït Aoudia, Jakob Hoydis, Merlin Nimier-David, Sebastian Cammerer, and Alexander Keller

Abstract

Sionna™ is an open-source, GPU-accelerated library that, as of version 0.14, incorporates a ray tracer for simulating radio wave propagation. A unique feature of Sionna RT is differentiability, enabling the calculation of gradients for the channel impulse responses (**CIRs**), radio maps, and other related metrics with respect to system and environmental parameters, such as material properties, antenna patterns, and array geometries. The release of Sionna 1.0 provides a complete overhaul of the ray tracer, significantly improving its speed, memory efficiency, and extensibility. This document details the algorithms employed by Sionna RT to simulate radio wave propagation efficiently, while also addressing their current limitations. Given that the computation of **CIRs** and radio maps requires distinct algorithms, these are detailed in separate sections. For **CIRs**, Sionna RT integrates **shooting and bouncing of rays (SBR)** with the image method and uses a hashing-based mechanism to efficiently eliminate duplicate paths. Radio maps are computed using a purely **SBR**-based approach.

Contents

1	Introduction	4
2	Essential Concepts and Terminology	6
2.1	Scene Objects and Meshes	6
2.2	Rays and Paths	7
2.3	Interactions with Scene Objects	7
2.4	Ray Tubes	8
3	Path Solver	8
3.1	Generating Candidates by Shooting and Bouncing of Rays (SBR)	10
3.1.1	Sampling Initial Ray Directions	13
3.1.2	Sampling Interaction Types	14
3.1.3	Hashing of Specular Chains	15
3.1.4	Potential Improvements	17
3.2	Image Method-based Candidate Processing	18
3.3	Channel Coefficients, Delays, and Doppler Shifts Computation	20
3.3.1	Handling Multiple Antennas at the Transmitter and Receiver	23
3.3.2	Doppler Shifts	23
4	Radio Map Solver	24
4.1	Definition of a Radio Map	24
4.1.1	Non-Coherent vs. Coherent Radio Maps	28
4.1.2	Handling Multiple Transmit Antennas	28
4.2	SBR-based Radio Map Solver	29
A	Primer on Electromagnetism	32
A.1	Coordinate System, Rotations, and Vector Fields	32
A.2	Planar Time-Harmonic Waves	33
A.3	Far Field of a Transmitting Antenna	34
A.4	Modeling of a Receiving Antenna	35
A.5	General Propagation Path	36
A.6	Frequency and Impulse Response	37
A.7	Specular Reflection and Refraction	37
A.7.1	Single-Layer Slab	39
A.8	Diffuse Reflection	40

List of Acronyms

BSDF bidirectional scattering distribution function.

CIR channel impulse response.

EM electromagnetic.

GCS global coordinate system.

JIT just-in-time.

LoS line-of-sight.

RIS reflective intelligent surface.

SBR shooting and bouncing of rays.

1. Introduction

Ray tracing for simulating radio wave propagation has received renewed attention in the recent years, fueled by the growing interest in creating digital twins for wireless communication systems and research topics that require simulating spatially consistent **CIRs**. In 2023, we released Sionna RT [1], the world's first fully differentiable ray tracer for radio propagation modeling, as part of version 0.14 of the Sionna open-source library for research in communication systems [2]. Over the following years, we have expanded the capabilities of Sionna RT with additional features, including support for diffraction and diffuse reflection (version 0.15), mobility (version 0.17), and **reflective intelligent surfaces (RISs)** (version 0.18).

Sionna RT is built on top of the **just-in-time (JIT)** compiler Dr.Jit [3] and the differentiable physically-based renderer Mitsuba [4]. Sionna RT implements ray tracing-based algorithms to model the propagation of **electromagnetic (EM)** radio waves, enabling the computation of **CIRs** and radio maps (also known as *coverage maps* or *power maps*). Additionally, it includes radio material models to simulate the interaction of radio waves with scatterers in a radio propagation environment. The release of Sionna 1.0 marks a significant milestone as it features a complete revision of the ray tracing module.¹ The ray tracer is now interoperable with major deep learning frameworks, including TensorFlow [5] and PyTorch [6]. Moreover, the computation of **CIRs** and radio maps has been substantially accelerated while maintaining full differentiability of the ray tracing process by leveraging the automatic differentiation capabilities of Dr.Jit. The ray tracer has also been made easier to extend to facilitate research on various aspects of radio propagation and digital twins.

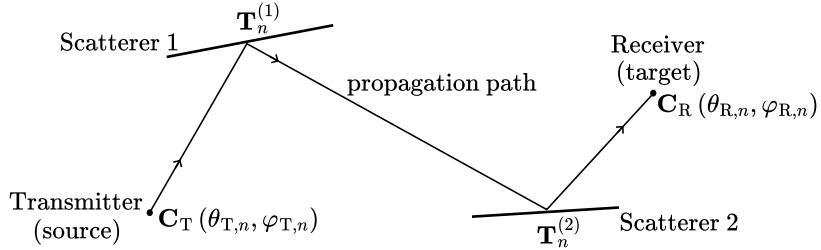


Figure 1: A propagation path, shown in 2D for clarity.

The primary objective of Sionna RT is to identify the *propagation paths* that link radio devices within a given propagation environment. As depicted in Figure 1, a propagation environment is defined by a collection of scatterers. A radio device functions as either a *transmitter*, which emits radio waves, or a *receiver*, which captures them, and is equipped with one or more antennas. The *channel frequency response* $H(f)$ characterizes the propagation of radio waves from a transmitting antenna to a receiving antenna for the frequency f , and is expressed as the ratio of the received voltage to the voltage at the input of the transmitting antenna:

$$H(f) = \frac{V_R}{V_T}. \quad (1)$$

As explained in Appendix A, the channel frequency response is the Fourier transform of the **CIR**, which is defined as:

$$h(\tau) = \sum_{n=1}^N a_n \delta(\tau - \tau_n). \quad (2)$$

Here, N represents the number of *propagation paths* connecting the transmit antenna to the receive antenna. The term a_n is the *complex-valued path coefficient* for the n -th path, and $\tau_n = r_n/c$ is the *propagation delay*, where r_n is the distance the radio wave travels, and c is the speed of light. This leads to:

$$H(f) = \int_{-\infty}^{\infty} h(\tau) e^{-j2\pi f \tau} d\tau = \sum_{n=1}^N a_n e^{-j2\pi f \tau_n}. \quad (3)$$

Notably, the path coefficient a_n is calculated by considering the sequence of matrices that model the transformations caused by interactions with scatterers along the n -th propagation path, as well as the characteristics of

¹As of version 1.0.2, Sionna RT does not currently support diffraction and **RISs**. These features are planned for future updates. In the meantime, they remain available as part of Sionna version 0.19.2.

the transmit and receive antennas:

$$a_n = \frac{\lambda}{4\pi} \mathbf{C}_R(\theta_{R,n}, \varphi_{R,n})^H \left(\prod_{\ell'=1}^{\ell} \mathbf{T}_n^{(\ell')} \right) \mathbf{C}_T(\theta_{T,n}, \varphi_{T,n}). \quad (4)$$

Here, λ is the wavelength, ℓ is the number of scatterers along the n -th path, \mathbf{C}_T is the complex-valued vector of dimension 2 representing the *transmit antenna pattern* evaluated for the angles of departure of the path $(\theta_{T,n}, \varphi_{T,n})$, \mathbf{C}_R is the complex-valued vector of dimension 2 representing the *receive antenna pattern* evaluated for the angles of arrival of the path $(\theta_{R,n}, \varphi_{R,n})$, and $\mathbf{T}_n^{(\ell')}$ is the 2×2 complex-valued matrix modeling the interaction with the ℓ' -th scatterer. Further details on the computation of the path coefficient are provided in Section 3. The corresponding *channel gain* is

$$g = \sum_{n=1}^N |a_n|^2. \quad (5)$$

It is crucial to understand that calculating the path coefficients and delays necessitates first determining the intersection points of the paths with the scene geometry, referred to as the *path vertices*. This is because computing the path coefficients (4) requires the knowledge of the geometrical features of the paths. Specifically, the antenna patterns are evaluated at the angles of departure and arrival of the paths, and computing the matrices $\mathbf{T}_n^{(\ell')}$ requires the angles of incidence and scattering of waves at the scatterers as well as the traveled distances. Once the path vertices have been determined, the path coefficients and delays are computed, considering the characteristics of the antennas and scatterers' materials.

Paths are determined between two endpoints: a *source* and a *target*. Ideally, paths would be traced between every pair of transmitter and receiver antennas, designating the sources and targets as the set of transmit and receive antennas, respectively. However, when radio devices are equipped with a large number of antennas, computing paths for all antenna pairs becomes computationally demanding. To mitigate this issue, Sionna RT introduces a *synthetic array* feature, which calculates paths only between radio devices instead of each antenna pair. Phase shifts due to the array geometry are then applied synthetically, as detailed in Section 3. Note that the use of synthetic arrays can be disabled. Throughout this document, the endpoints of a path will be referred to as sources and targets, which can be either radio devices or antennas. Although the use of synthetic arrays does not change the pathfinding process, it does affect the computation of the path coefficient and delays.

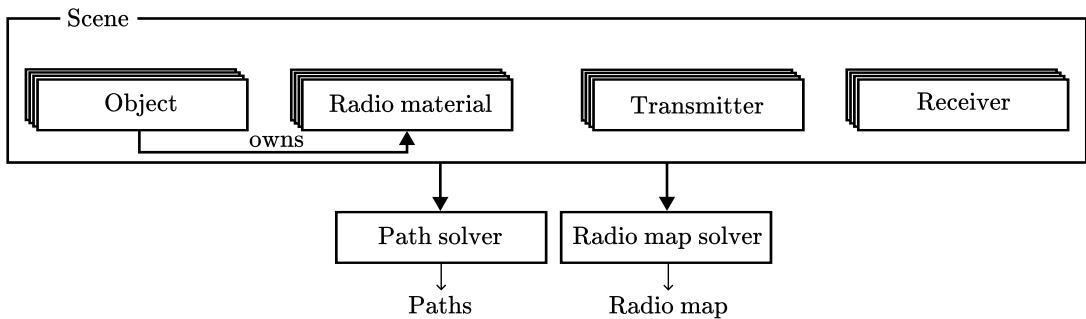


Figure 2: Sionna RT architecture overview.

Figure 2 illustrates the overall architecture of Sionna RT. A *scene* represents a radio propagation environment, comprising a set of *objects*, *radio materials*, and *radio devices*. Each object acting as a scatterer is associated with a radio material that determines its interaction with incident radio waves. Multiple objects can share the same radio material. Radio materials are akin to *bidirectional scattering distribution function (BSDF)* in the field of computer graphics, see e.g. [7, Ch. 4.3.1]. Note that an *object* alone defines only the geometry of a scatterer and does not contain any information about the radio material. Radio devices are either transmitters or receivers. A *solver* implements an algorithm for simulating radio wave propagation within a scene. Given the distinct algorithmic requirements for *CIRs* and radio maps, Sionna RT provides two types of solvers:

- The *path solver* computes a set of paths connecting a source to a target, which can then be used to compute a *CIR*.

- The *radio map solver* computes a radio map, a 2D grid of values that approximate the channel gain (5) that a receiver located on the grid would observe.

While Sionna RT provides built-in solvers and radio material models, it is designed to facilitate the implementation of custom alternatives.

Contributions of the Technical Report

Beyond serving as a technical reference, this technical report provides theoretical contributions by detailing the key methods implemented in Sionna RT for simulating radio wave propagation by ray tracing. For computing paths, a method that combines **SBR** with the image method is presented. Although this approach existed in earlier versions of Sionna RT, it has been significantly refined in Sionna 1.0, notably by incorporating a hashing-based mechanism to efficiently remove duplicate paths. The technical report also introduces a formal definition of radio maps, which forms the foundation for the radio map solver implemented in Sionna RT, as it enables efficient computation through **SBR** alone. It is worth noting that **SBR** is already extensively used in the graphics community (see e.g. [7]). Moreover, we include discussions on the limitations of the current algorithms and offer perspectives for future improvements.

Structure of the Technical Report

Section 2 introduces the essential concepts and terminology. Sections 3 and 4 provide an in-depth look at the built-in path solver and radio map solver of Sionna RT, respectively. These sections primarily focus on path tracing algorithms, specifically the methods used to determine the intersection points of paths with scene geometry. Appendix A provides a primer on **EM** theory and explains the computation of the electric field along a path.

2. Essential Concepts and Terminology

2.1. Scene Objects and Meshes

A scene object is represented as a *mesh*, which is composed of a set of triangles, also known as *primitives*, connected by their edges and vertices, as shown in Figure 3. Each object is identified by an index $o \in \mathbb{N}_0$, and each primitive within an object is identified by an index $m \in \mathbb{N}_0$. Primitives from different objects can share the same indices, so a primitive is uniquely identified by the pair $(o, m) \in \mathbb{N}_0^2$.

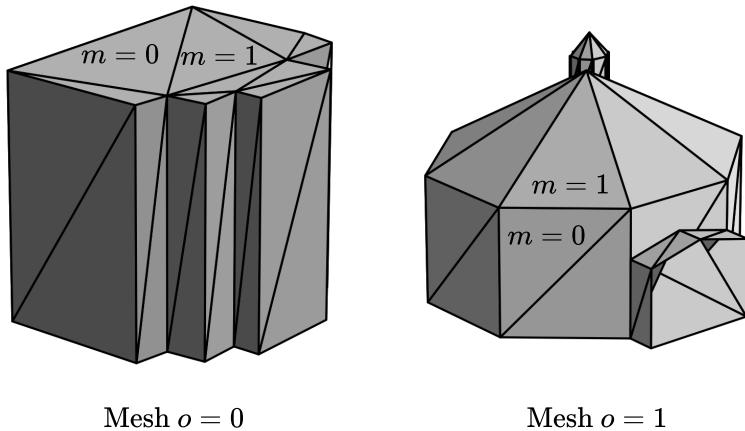


Figure 3: A mesh is composed of triangular primitives connected by their edges and vertices. For clarity, only the indices of two primitives are displayed for each mesh. These meshes were downloaded from OpenStreetMap [8] using Blosm [9].

2.2. Rays and Paths

A *ray* $(\mathbf{v}, \hat{\mathbf{k}}, t)$ is characterized by an origin $\mathbf{v} \in \mathbb{R}^3$, a direction $\hat{\mathbf{k}} \in \mathbb{R}^3$, and a length $t \geq 0$. The direction $\hat{\mathbf{k}}$ is a unit vector, meaning $\|\hat{\mathbf{k}}\|_2 = 1$. If the length is unspecified, the ray is considered infinite, i.e. $t = \infty$. A *path* $p^{(L)}$ with *depth* $L \in \mathbb{N}$ is a sequence of $L + 1$ rays that connect a *source point* $\mathbf{s} \in \mathbb{R}^3$ to a *target point* $\mathbf{t} \in \mathbb{R}^3$, as illustrated in Figure 4,

$$p^{(L)} = \left(\underbrace{(\mathbf{v}^{(0)}, \hat{\mathbf{k}}^{(0)}, t^{(0)})}_{\text{ray 0}}, \dots, \underbrace{(\mathbf{v}^{(L)}, \hat{\mathbf{k}}^{(L)}, t^{(L)})}_{\text{ray } L} \right) \quad (6)$$

where the superscript (L) indicates the depth of the path, $\mathbf{v}^{(0)} = \mathbf{s}$, $\mathbf{v}^{(i+1)} = \mathbf{v}^{(i)} + t^{(i)}\hat{\mathbf{k}}^{(i)}$, and $\mathbf{t} = \mathbf{v}^{(L)} + t^{(L)}\hat{\mathbf{k}}^{(L)}$. The origins of the rays that form a path starting at the source and the target are known as the *vertices* of the path. Hence, a path may be as well defined by a sequence of $L + 2$ vertices:

$$p^{(L)} = (\mathbf{v}^{(0)} = \mathbf{s}, \mathbf{v}^{(1)}, \dots, \mathbf{v}^{(L)}, \mathbf{v}^{(L+1)} = \mathbf{t}) \quad (7)$$

In this case, the rays have directions $\hat{\mathbf{k}}^{(i)} = \frac{\mathbf{v}^{(i+1)} - \mathbf{v}^{(i)}}{\|\mathbf{v}^{(i+1)} - \mathbf{v}^{(i)}\|_2}$ and lengths $t^{(i)} = \|\mathbf{v}^{(i+1)} - \mathbf{v}^{(i)}\|_2$ for $0 \leq i \leq L$.

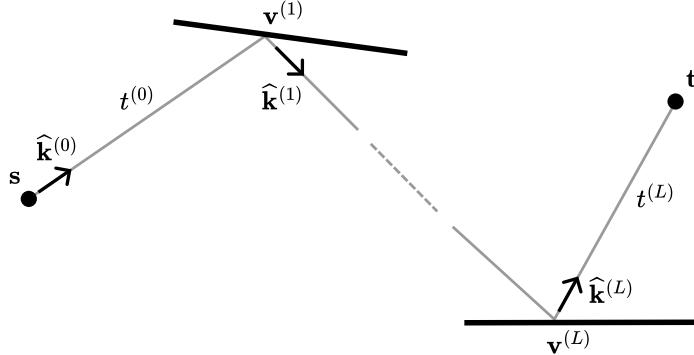


Figure 4: A path consists of a sequence of ray segments connecting a source point \mathbf{s} to a target point \mathbf{t} .

The *suffix* of a path, denoted by $p^{(\ell:)} \text{ where } 0 \leq \ell \leq L$, is defined as the sub-path consisting of all the rays from the ℓ -th one onward, i.e.

$$p^{(\ell:)} = ((\mathbf{v}^{(\ell)}, \hat{\mathbf{k}}^{(\ell)}, t^{(\ell)}), \dots, (\mathbf{v}^{(L)}, \hat{\mathbf{k}}^{(L)}, t^{(L)})) . \quad (8)$$

Similarly, the corresponding *prefix* is:

$$p^{(:\ell)} = ((\mathbf{v}^{(0)}, \hat{\mathbf{k}}^{(0)}, t^{(0)}), \dots, (\mathbf{v}^{(\ell-1)}, \hat{\mathbf{k}}^{(\ell-1)}, t^{(\ell-1)})) . \quad (9)$$

2.3. Interactions with Scene Objects

Sionna RT currently supports three types of interactions with scene objects:

Specular reflection (\mathcal{R}): The wave is reflected with a reflection angle equal to the incident angle, as illustrated in Figure 5a.

Diffuse reflection (\mathcal{S}): The wave is reflected in multiple directions, as illustrated in Figure 5a.

Refraction (\mathcal{T}): The wave propagates into the scattering medium. The solvers assume that object surfaces are thin enough that their effect on transmitted rays (i.e. rays that traverse the surfaces through double refraction) can be modeled by a single transmitted ray. The transmitted rays are traced without angular deflection. Surfaces like walls should be modeled as single flat surfaces, as illustrated in Figure 5b. However, when computing the transmitted and reflected fields, the thickness of the traversed object is considered.

The set of possible interaction types is denoted by $\mathcal{I} = \{\mathcal{R}, \mathcal{S}, \mathcal{T}\}$.

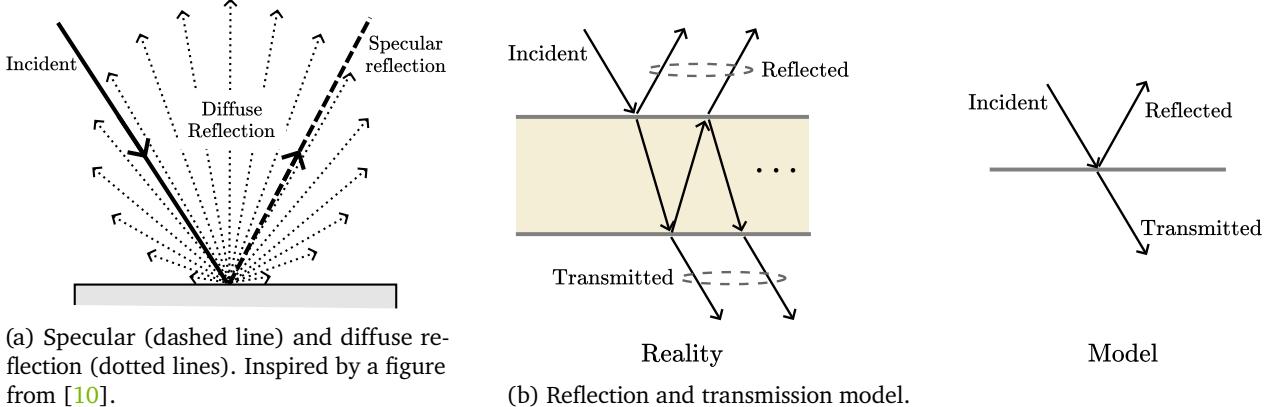


Figure 5: Sionna RT currently supports specular reflection, diffuse reflection, and refraction.

2.4. Ray Tubes

While a more detailed background on the propagation of **EM** waves is provided in Appendix A, we introduce here some essential **EM** concepts that are required for understanding the process of ray tracing. The source s is modeled as a *point source*, an infinitesimally small emitter of **EM** waves that radiates in all directions according to a user-defined *transmitter antenna pattern*. Note that modeling transmit antennas as point sources is a valid approximation when the distances to scatterers and targets are significantly greater than the wavelength of the propagating waves. Each ray traced from the source s serves as the axial ray of a *ray tube* [11, Chapter 2], which is a bundle of rays adjacent to the axial one. These ray tubes originate at the source and have a length r and an angular opening ω , as illustrated in Figure 6. Importantly, specular reflection and reflection through planar surfaces only alter the ray direction. Diffuse reflections alter the shape of the wavefront as they spawn new point sources. As detailed in Sections 3 and 4, this process requires specific computations to determine the path vertices and corresponding fields. For a given path $p^{(L)}$ with depth L , we denote by $\chi^{(L)} = (\chi^{(1)}, \dots, \chi^{(L)}) \in \mathcal{I}^L$ the sequence of interaction types along the path. A key quantity for the remainder of this document is the depth of the last diffuse reflection, denoted by $\ell_d \in \mathbb{N}$ and defined as:

$$\ell_d = \begin{cases} \max_{1 \leq \ell \leq L} \{\ell : \chi^{(\ell)} = \mathcal{S}\} & \text{if } \exists \ell : \chi^{(\ell)} = \mathcal{S} \\ 0 & \text{otherwise.} \end{cases} \quad (10)$$

The dependency of ℓ_d on $\chi^{(L)}$ is omitted for brevity. The case where $\ell_d = 0$ corresponds to paths that do not contain any diffuse reflection, known as *specular chains*. If $\ell_d > 0$, then the suffix $p^{(\ell_d:)}_d$ is referred to as the *specular suffix* of p . The value ℓ_d represents the depth index from which the path is composed solely of specular reflections and refractions, and is termed the *specular suffix index* of p .

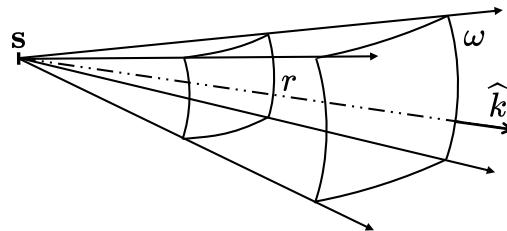


Figure 6: A ray tube is a bundle of rays adjacent to an axial ray.

3. Path Solver

A path solver aims to determine a set of paths that connect two endpoints in a scene, as illustrated in Figure 7. From this set of paths, a time- or frequency-domain **CIR** can be computed. The path solver leverages **SBR**

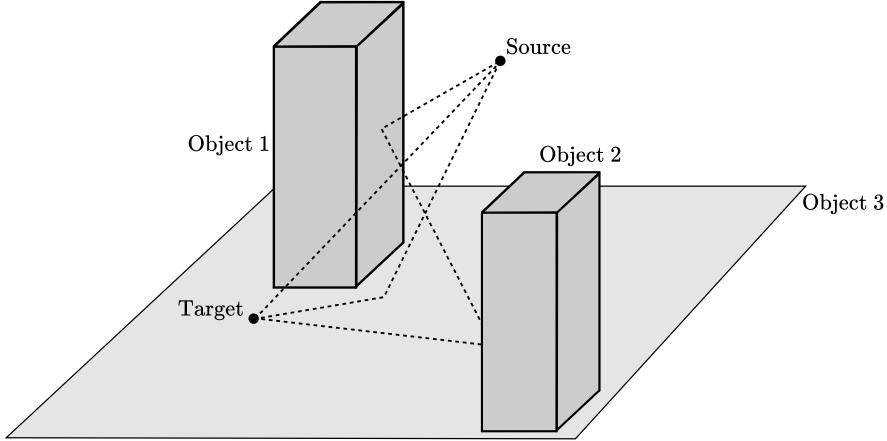


Figure 7: The path solver constructs propagation paths that connect a source and a target in a scene.

to efficiently compute paths in a scene. SBR operates as follows: First, rays² are traced from the source into directions specified by a lattice on the unit sphere (see Section 3.1.1). Then, an SBR loop iteratively tests the intersection of the rays with the scene geometry. At every intersection, an interaction type is selected from specular reflection, diffuse reflection, and refraction, and a new ray is spawned according to the selected interaction type. The method for choosing interaction types is detailed in Section 3.1.2. The loop terminates when no more rays remain *active*, meaning they have either bounced out of the scene, reached a predefined maximum number of bounces, or satisfied another termination criterion. Notably, at each interaction of a ray with a surface in the scene, only one scattered ray is spawned, as spawning multiple rays would result in an exponential increase in the number of rays, quickly overwhelming the available computational resources.

When a diffuse reflection occurs during the SBR loop, the solver checks whether there is a *line-of-sight* (LoS) to the target, meaning that the ray segment from the interaction point to the target point is not occluded. If such a connection exists, the path is deemed valid and is returned by the solver. This process is known as *next-event estimation*. Subsequently, a new ray is spawned in a random direction within the hemisphere defined by the surface normal of the primitive involved. This new ray may lead to the discovery of additional paths. Next-event estimation is only possible for diffuse reflections that scatter energy into all directions. For specular reflection and refraction, only a single ray is spawned in the only possible direction. Consequently, the probability of finding paths connecting the source to the target that end with a specular reflection or refraction by SBR is effectively zero, as depicted in Figure 8. This is because hitting a specific point (the target) with a specular reflection or refraction requires perfect alignment of the incident ray, which has zero probability in continuous space. Therefore, paths ending with a specular suffix cannot be determined solely by SBR. Especially, specular chains cannot be computed by SBR. Since specular chains typically carry a significant amount of the transported energy, SBR alone is insufficient for computing CIRs.

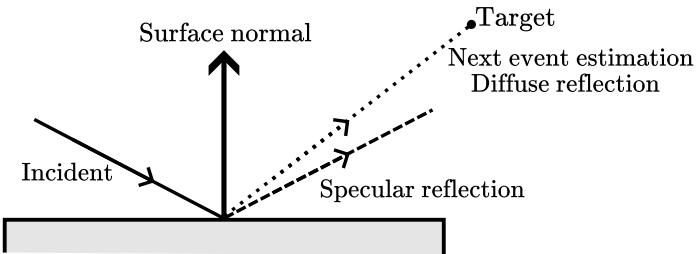


Figure 8: When a diffuse reflection occurs, next event estimation tests whether the LoS between the intersection point and the target is occluded (represented by the dotted line). This process is not applicable for specular reflection (represented by the dashed line) and refraction, as only one single direction for the scattered ray is valid.

²The processing of the rays is parallelized for accelerated computation.

To compute paths ending with a specular suffix, including specular chains, the path solver uses an *image method*-based algorithm. However, a brute-force implementation of the image method would test all possible combinations of primitives that constitute the scene and hence is prohibitive even for moderately large scenes. Therefore, the path solver uses **SBR** as an heuristic to find *path candidates*. A path candidate consists of a sequence of primitives and interaction types ending with a specular reflection or refraction. As the path vertices determined during the **SBR** do not form a path that connects to the target, the image method is used to adjust the vertex locations to establish the connection. Intuitively, **SBR** pre-selects path candidates in the vicinity of the transmitter and receiver for the image method to compute corresponding valid paths to be returned by the solver. These steps of the Sionna RT path solver are depicted in Figure 9. Notably, only path candidates ending with a specular suffix (including specular chains) require further processing by the image method algorithm. Finally, for each valid path identified, the path solver computes the corresponding channel coefficients and propagation delays that characterize the **EM** wave propagation along that path.

Alternative methods for identifying paths that include specular chains or specular suffixes may involve detection spheres centered at the targets to capture paths ending with a specular suffix, and/or extending the direction of specular reflection (or refraction) to a lobe around the specular (or refracted) direction. Such methods only approximate the path vertex locations and the directions of the scattered rays, yielding inaccurate path coefficients and consequently resulting in incorrect **CIRs**.

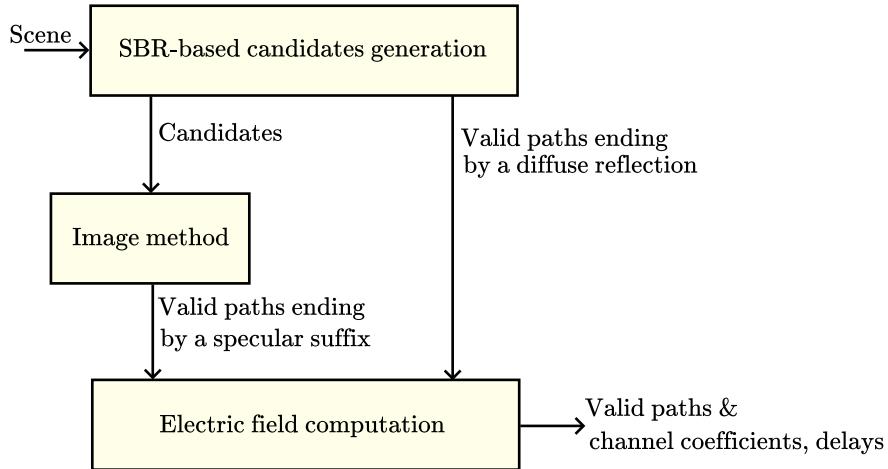


Figure 9: Main steps of the Sionna RT path solver.

The remainder of this section follows the architecture of the path solver shown in Figure 9. First, Section 3.1 presents the **SBR**-based candidate generator. Section 3.2 then describes the image method-based algorithm that processes candidates to find valid paths with specular suffixes. Finally, Section 3.3 details how the solver computes the complex-valued channel coefficients and propagation delays, which characterize the radio channel.

3.1. Generating Candidates by Shooting and Bouncing of Rays (SBR)

The candidate generator utilizes the **SBR**-based method as illustrated in Figure 10. We first provide an overview of the **SBR** procedure, followed by a detailed explanations. We consider a scene with a single source located at $s \in \mathbb{R}^3$ and N_T targets at $t_k \in \mathbb{R}^3$, where $k = 1, \dots, N_T$. Although the path solver built into Sionna RT can handle multiple sources, we focus on a single source for the sake of clarity. The process is identical for all sources. The paths traced by the candidate generator during the **SBR** procedure are referred to as *samples* in this section to avoid confusion with the valid paths returned to the user and the candidate paths forwarded to the image method-based algorithm. A sample therefore consists of a sequence of rays connected end-to-end, traced by the candidate generator. As we will see, a single sample can result in multiple valid or candidate paths, as illustrated in Figure 11.

The **SBR** procedure begins by spawning a user-specified number of rays, denoted by N_S , from the source s in directions $\hat{k}_n^{(0)}$ on a Fibonacci lattice, where $n = 1, \dots, N_S$. Each ray starts a sample that is traced by the

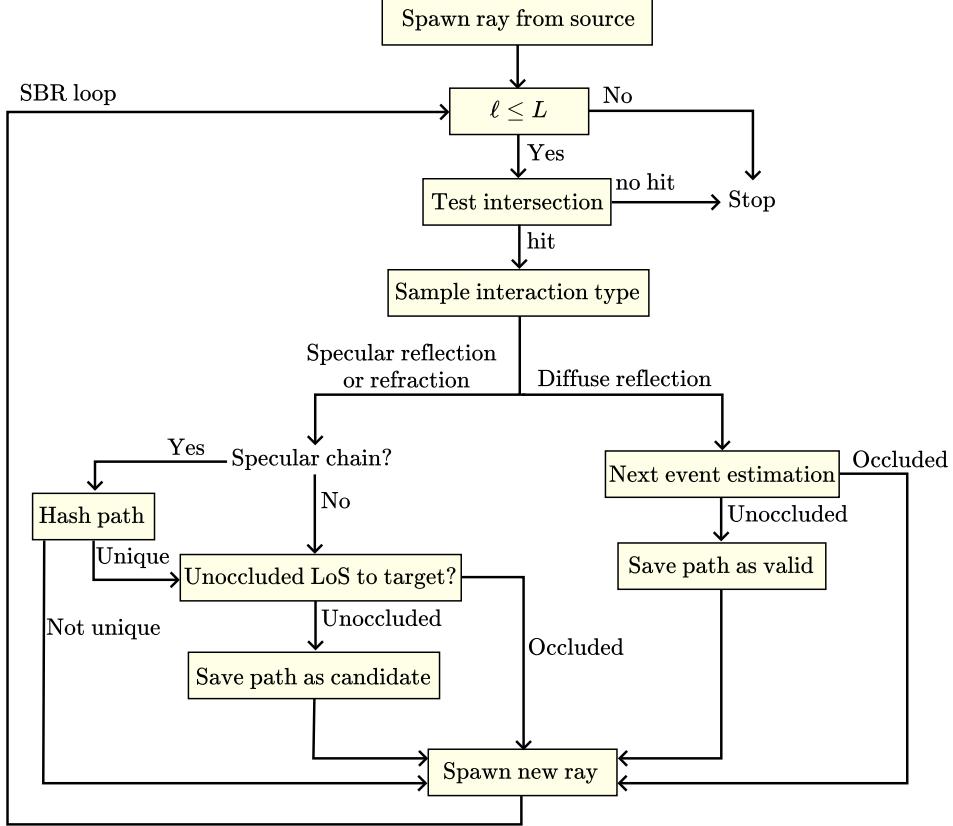


Figure 10: Algorithm for one sample as implemented in the **SBR**-based candidate generator. In practice, many samples are processed in parallel.

candidate generator. Details on the implemented Fibonacci lattice can be found in Section 3.1.1. The **SBR** loop then iterates until either all samples bounce out of the scene or reach a user-specified maximum number of bounces, denoted as the *maximum depth* L . We denote the depth of the samples by $\ell = 0, \dots, L$. The first step of the **SBR** loop involves testing the intersection of the samples with the scene geometry. If a sample does not intersect any object, it has bounced out of the scene, and is deactivated. When a sample n with depth ℓ intersects the scene, we denote by $\mathbf{v}_n^{(\ell)} \in \mathbb{R}^3$ the intersection point, by $\hat{\mathbf{n}}_n^{(\ell)} \in \mathbb{R}^3$ ($\|\hat{\mathbf{n}}_n^{(\ell)}\|_2 = 1$) the surface normal at the intersection point oriented towards the half-space containing the incident field, by $o_n^{(\ell)} \in \mathbb{N}_0$ the index of the intersected object, and by $m_n^{(\ell)} \in \mathbb{N}_0$ the index of the specific intersected primitive within that object. An interaction type, denoted by $\chi_n^{(\ell)}$, is then sampled among specular reflection, diffuse reflection, and refraction, which determines how the sample will continue to propagate. Details on the sampling of the interaction type are provided in Section 3.1.2. The rest of the loop depends on the sampled interaction type.

Diffuse Reflection

As previously indicated, diffuse reflection is assumed to scatter energy in all directions of the half-space containing the incident field, which enables next event estimation. Therefore, if a diffuse reflection is sampled (i.e. $\chi_n^{(\ell)} = \mathcal{S}$), the **LoS** between the intersection point and each target $\mathbf{t}_1, \dots, \mathbf{t}_{N_T}$ is tested. For each target \mathbf{t}_k with an unobstructed **LoS** to the intersection point, a path connecting the source to the target

$$p_n^{(\ell)} = \left(\mathbf{s}, \left(\mathbf{v}_n^{(1)}, o_n^{(1)}, m_n^{(1)}, \chi_n^{(1)} \right), \dots, \left(\mathbf{v}_n^{(\ell)}, o_n^{(\ell)}, m_n^{(\ell)}, \chi_n^{(\ell)} \right), \mathbf{t}_k \right) \quad (11)$$

is recorded as a valid path to be returned to the user. Note that the path here is enriched with the intersected primitives and the interaction types. Finally, a new ray is spawned to continue the sample path, with its direction selected uniformly at random from the hemisphere defined by $\hat{\mathbf{n}}_n^{(\ell)}$, i.e. $\hat{\mathbf{k}}_n^{(\ell+1)} \sim \mathcal{U}_{2\pi}(\hat{\mathbf{n}}_n^{(\ell)})$, where $\mathcal{U}_{2\pi}(\hat{\mathbf{n}}_n^{(\ell)})$ represents the uniform distribution on the hemisphere.

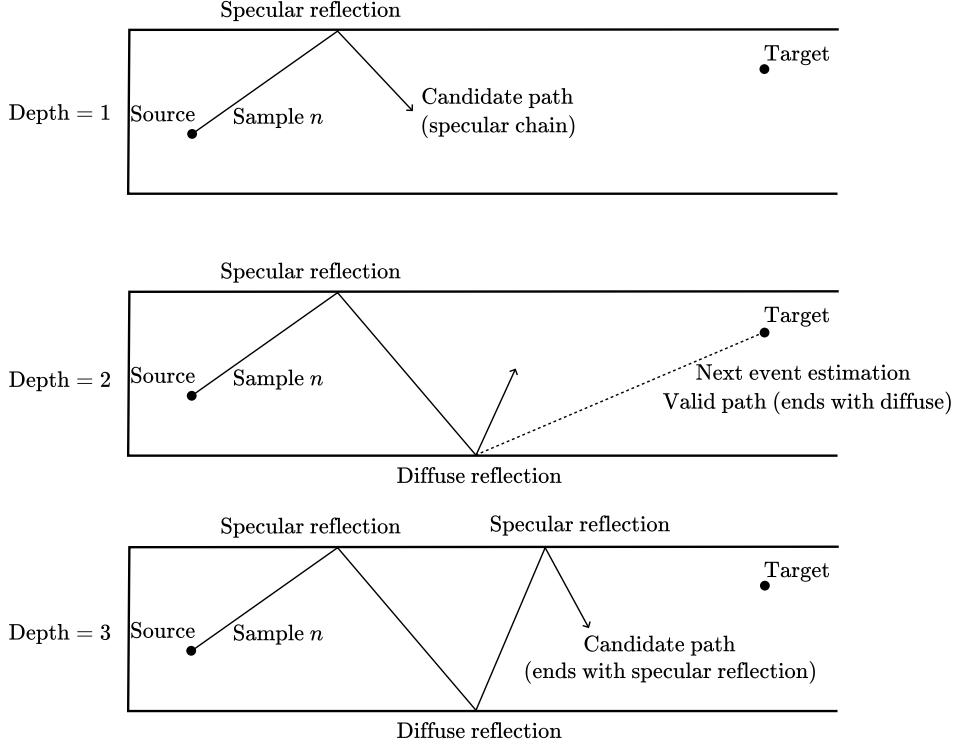


Figure 11: SBR procedure for a single sample n at depths 1, 2, and 3. At each interaction (i.e. iteration of the SBR loop), the sample may result in a candidate or valid path, and a new ray is spawned from the current interaction point to prolong the sample path. Shown in 2D for clarity.

Specular Reflection and Refraction

When a specular reflection or refraction is sampled (i.e. $\chi_n^{(\ell)} = \mathcal{R}$ or $\chi_n^{(\ell)} = \mathcal{T}$), the path (11) is a valid path for target \mathbf{t}_k , where $k \in \{1, \dots, N_T\}$, only if

$$\frac{\mathbf{t}_k - \mathbf{v}_n^{(\ell)}}{\|\mathbf{t}_k - \mathbf{v}_n^{(\ell)}\|_2} = \hat{\mathbf{k}}_n^{(\ell+1)}, \quad (12)$$

where

$$\hat{\mathbf{k}}_n^{(\ell+1)} = \begin{cases} \hat{\mathbf{k}}_n^{(\ell)} - 2 \left((\hat{\mathbf{k}}_n^{(\ell)})^\top \hat{\mathbf{n}}_n^{(\ell)} \right) \hat{\mathbf{n}}_n^{(\ell)} & \text{if } \chi_n^{(\ell)} = \mathcal{R} \\ \hat{\mathbf{k}}_n^{(\ell)} & \text{if } \chi_n^{(\ell)} = \mathcal{T} \end{cases} \quad (13)$$

and where \mathbf{u}^\top denotes the transpose of \mathbf{u} . Since targets are modeled as points, condition (12) effectively occurs with zero probability, and, consequently, the path (11) is not a valid path. However, the specular suffix $p_n^{(\ell_d)}$, where ℓ_d is defined as in (10), can be further processed to determine vertices $(\tilde{\mathbf{v}}_n^{(\ell_d+1)}, \dots, \tilde{\mathbf{v}}_n^{(\ell)})$ such that

$$\begin{aligned} \tilde{p}_n^{(\ell)} = & \left(\mathbf{s}, (\mathbf{v}_n^{(1)}, o_n^{(1)}, m_n^{(1)}, \chi_n^{(1)}), \dots, \right. \\ & \left. \underbrace{(\tilde{\mathbf{v}}_n^{(\ell_d+1)}, o_n^{(\ell_d+1)}, m_n^{(\ell_d+1)}, \chi_n^{(\ell_d+1)}), \dots, (\tilde{\mathbf{v}}_n^{(\ell)}, o_n^{(\ell)}, m_n^{(\ell)}, \chi_n^{(\ell)}), \mathbf{t}_k}_{\text{Processed specular suffix}} \right) \end{aligned} \quad (14)$$

is a valid path, as illustrated in Figure 12. This additional processing is performed using the image method in the algorithm described in Section 3.2.

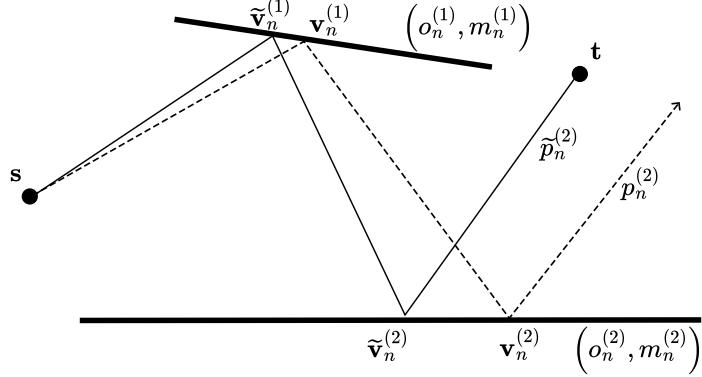


Figure 12: Processing a candidate specular chain of depth $\ell = 2$ ($p_n^{(2)}$, dashed line) to produce a valid path ($\tilde{p}_n^{(2)}$, solid line). The same process applies to specular suffixes. Shown in 2D for clarity.

Importantly, for a sequence of primitives and interaction types

$$\left(\mathbf{v}_n^{(\ell_d)}, \left(o_n^{(\ell_d+1)}, m_n^{(\ell_d+1)}, \chi_n^{(\ell_d+1)}\right), \dots, \left(o_n^{(\ell)}, m_n^{(\ell)}, \chi_n^{(\ell)}\right), \mathbf{t}_i\right) \quad (15)$$

where $\mathbf{v}_n^{(0)} = \mathbf{s}$, there exists *at most a single* set of vertices $\tilde{\mathbf{v}}_n^{(\ell_d+1)}, \dots, \tilde{\mathbf{v}}_n^{(\ell)}$ such that $\tilde{p}_n^{(\ell)}$ is a valid path. Consequently, if $\ell_d = 0$, i.e. $p_n^{(\ell)}$ is a specular chain, then further processing by the image method of multiple identical candidates (15) would result in multiple identical valid paths, which would lead to erroneous CIRs. Therefore, if the sample $p_n^{(\ell)}$ is a specular chain, a hashing-based de-duplication mechanism, detailed in Section 3.1.3, ensures that specular chain candidates are only evaluated once. Notably, the path hashing mechanism discards redundant candidates “on-the-fly”, significantly reducing memory and compute requirements. However, when $\ell_d > 0$, then $\mathbf{v}_n^{(\ell_d)}$ corresponds to a diffuse reflection point that is the result of a randomly sampled ray direction (either at the source or the last previous diffuse reflection point). Since the probability of two different paths having identical randomly sampled diffuse reflection points is effectively zero, path uniqueness is guaranteed in these cases without requiring the de-duplication mechanism. As shown in Figure 10, a LoS occlusion test to the target is used as a heuristic to discard path candidates that have little chance of resulting in a valid path after processing by the image method-based algorithm. Finally, a new ray is traced into the direction (13) to continue the sample path.

The candidate generator depicted in Figure 10 thus yields a set of valid paths connecting the source to all targets that do not require further processing (except for the computation of the corresponding path coefficients and delays) and end with a diffuse reflection. Additionally, it produces a set of candidate paths ending with specular suffixes, which require further processing to be either discarded or refined into valid paths. The path solver constructs paths that may include any combination and number of specular reflections, diffuse reflections, and refractions.

3.1.1. Sampling Initial Ray Directions

The SBR procedure starts by spawning rays from the source with directions obtained from a spherical Fibonacci lattice with N_S points, which is defined as follows:

$$\hat{\mathbf{k}}_n^{(0)} = \begin{bmatrix} \sin \theta_n \cos \phi_n \\ \sin \theta_n \sin \phi_n \\ \cos \theta_n \end{bmatrix} \quad (16)$$

where

$$\begin{aligned} \theta_n &= \arccos \frac{2n}{N_S} \\ \phi_n &= 2\pi \frac{n}{\varphi} \end{aligned}$$

and $-\lfloor \frac{N_S}{2} \rfloor \leq n \leq \lceil \frac{N_S}{2} \rceil - 1$, $\varphi = \frac{1+\sqrt{5}}{2}$ is the golden ratio, and $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ the floor and ceiling function, respectively. Note that when defining the Fibonacci lattice, the index n takes values in the range $-\lfloor \frac{N_S}{2} \rfloor, \dots, \lceil \frac{N_S}{2} \rceil - 1$ instead of the usual $1, \dots, N_S$. Sampling rays from the Fibonacci lattice results in ray tubes sharing approximately the same opening angle $\frac{4\pi}{N_S}$.

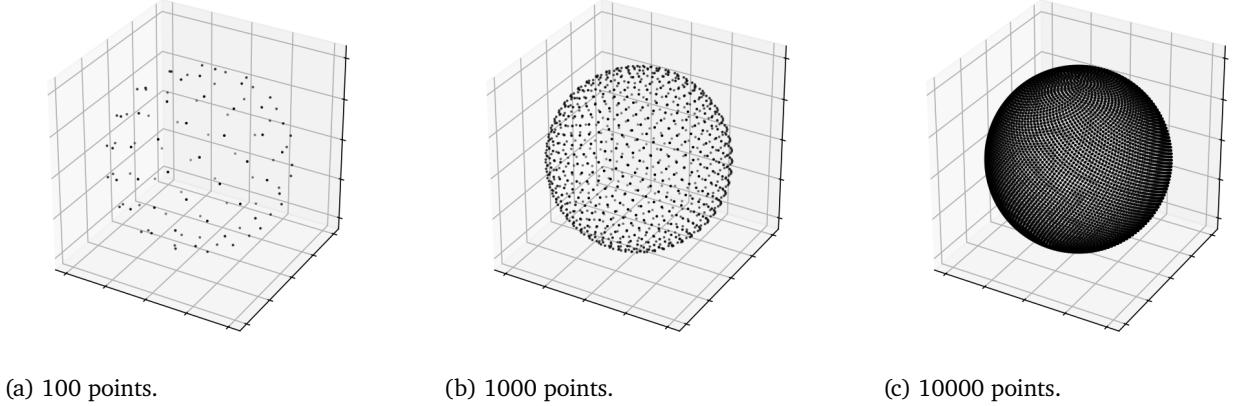


Figure 13: Spherical Fibonacci lattice.

The Fibonacci lattice was selected for its simple construction and its benefits over other lattices for numerical integration [12, 13]. Figure 13 shows the spherical Fibonacci lattice with 100, 1000, and 10000 points. The Sionna RT path solver uses 10^6 points as the default value for N_S , and higher values will ensure that more valid paths are found.

3.1.2. Sampling Interaction Types

Within the **SBR** loop, when a ray intersects the scene geometry, the algorithm generates at most one scattered ray. This constraint is necessary because generating multiple rays at each intersection, such as both reflected and refracted rays, would lead to an exponential increase in computational complexity and memory usage with the depth of the path. Instead, a single interaction type is randomly chosen based on a distribution:

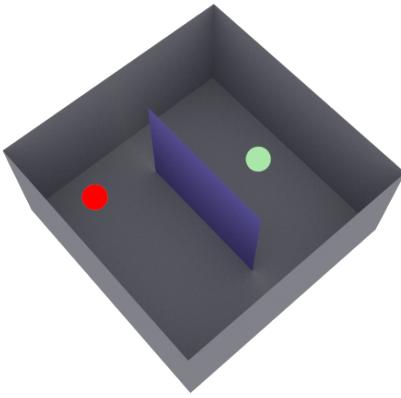
$$\mathcal{Q} = \{q(\mathcal{R}), q(\mathcal{S}), q(\mathcal{T})\}, \quad \text{where } q(\mathcal{R}) + q(\mathcal{S}) + q(\mathcal{T}) = 1 \text{ and } q(\cdot) \geq 0. \quad (17)$$

Here, $q(\chi)$ represents the probability of selecting the interaction type $\chi \in \mathcal{I}$. Importantly, the choice of distribution \mathcal{Q} does not affect the correctness of the calculated paths, which are solely dependent on the propagation environment. However, the choice of distribution \mathcal{Q} significantly impacts the *sample efficiency* of the ray tracer, which is the number of samples N_S needed to compute paths that connect a source to a target and capture the majority of the transported energy. Sampling only one interaction type per intersection may appear limiting, but this is offset by the large number of samples generated from each source. In most scenarios, a high sample count allows for identifying all significant paths.

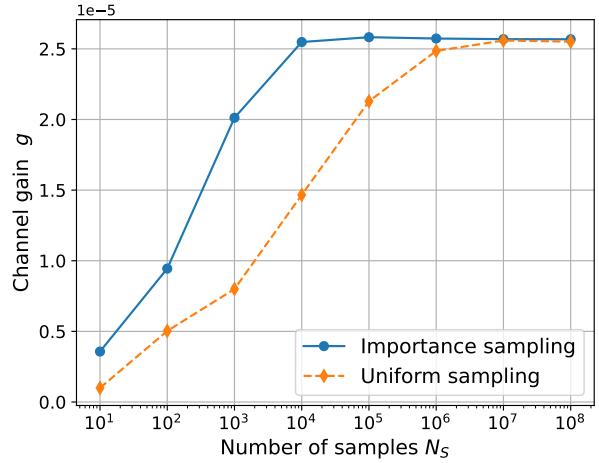
For each sample n at depth ℓ , the interaction type is sampled independently from other interactions and such that the probability of sampling each event type is proportional to the squared amplitudes of the reflection and refraction coefficients:

$$\begin{aligned} q(\mathcal{R}) &= R^2 \frac{|r_{\perp}|^2 + |r_{\parallel}|^2}{|r_{\perp}|^2 + |r_{\parallel}|^2 + |t_{\perp}|^2 + |t_{\parallel}|^2} \\ q(\mathcal{S}) &= S^2 \frac{|r_{\perp}|^2 + |r_{\parallel}|^2}{|r_{\perp}|^2 + |r_{\parallel}|^2 + |t_{\perp}|^2 + |t_{\parallel}|^2} \\ q(\mathcal{T}) &= \frac{|t_{\perp}|^2 + |t_{\parallel}|^2}{|r_{\perp}|^2 + |r_{\parallel}|^2 + |t_{\perp}|^2 + |t_{\parallel}|^2} \end{aligned} \quad (18)$$

Here, r_{\perp} , r_{\parallel} , t_{\perp} , and t_{\parallel} are the Fresnel coefficients (116), S is the scattering coefficient (see Section A.8), and R is the reflection reduction factor (119). This approach ensures that the interaction types that scatter



(a) Scene used for the comparison.



(b) Channel gain at the target.

Figure 14: Comparison of sampling strategies for interaction types. The scene consists of a closed metallic box containing a glass screen, with a source (red ball) and target (green ball) on opposite sides. The box is shown open to visualize the interior. Importance sampling achieves faster convergence of the channel gain compared to uniform sampling.

more energy are sampled more frequently—a technique known as *importance sampling* [14]. This is similar to techniques for importance sampling of light transport paths as used in computer graphics.

To illustrate the importance of optimized sampling strategies, we compare uniform sampling, where each interaction type is selected with equal probability, to importance sampling using the scene shown in Figure 14a. The scene consists of a metallic box containing a glass screen, with a source and target positioned on opposite sides. Diffuse reflection, specular reflection, and refraction are enabled, and the maximum depth L is set to 5. As shown in Figure 14b, using importance sampling leads to better sample efficiency when compared to uniform sampling, as convergence requires approximately $100\times$ less samples to capture most of the received energy. This demonstrates the benefit of selecting interactions according to the material properties of the scatterers.

3.1.3. Hashing of Specular Chains

As mentioned in Section 3.1 and illustrated in Figure 10, to avoid storing multiple identical specular chain candidates, a path hashing method is used to discard duplicates of previously found candidates “on-the-fly”. Recall that a specular chain candidate is defined by a sequence of interactions with the scene, each characterized by the intersected object $o_n^{(\ell)} \in \mathbb{N}_0$, the corresponding intersected primitive $m_n^{(\ell)} \in \mathbb{N}_0$, and the interaction type $\chi_n^{(\ell)} \in \mathcal{I}$, as in (15) with $\ell_d = 0$, i.e.

$$c_n^{(\ell)} = \left(\mathbf{s}, \left(o_n^{(1)}, m_n^{(1)}, \chi_n^{(1)} \right), \dots, \left(o_n^{(\ell)}, m_n^{(\ell)}, \chi_n^{(\ell)} \right) \right). \quad (19)$$

For specular chain candidates, we do not need to store the path vertices. If the candidate leads to a valid path, these vertices will be computed. Invalid candidates will be discarded.

Algorithm 1 demonstrates how path hashing works during ray tracing to avoid duplicate specular chain candidates. It starts by initializing a hash value $\text{hash}_n^{(0)}$ to 0 for each sample $n = 1, \dots, N_S$ being traced. Then, as the sample interacts with objects in the scene, the algorithm computes a hash value $\text{inter_hash}_n^{(\ell)}$ for each intersection based on the object that was hit $o_n^{(\ell)}$, the specific primitive of that object $m_n^{(\ell)}$, and the type of interaction $\chi_n^{(\ell)}$. The overall path hash $\text{hash}_n^{(\ell)}$ gets updated by combining it with the hash of each new intersection, creating a unique identifier for the entire sample up to that point.

Computing the hash after a given intersection is implemented by combining the three components of an intersection into a single hash value through a two-step process using the *Cantor pairing function*. The

Algorithm 1 Hashing of specular chains

```

1:  $\text{hash}_n^{(0)} \leftarrow 0$ 
2:  $\ell \leftarrow 1$ 
3: while <Stop condition> do ▷ SBR loop
4:   ...
5:    $\text{inter\_hash}_n^{(\ell)} \leftarrow \text{HASH\_INTERSECTION}(o_n^{(\ell)}, m_n^{(\ell)}, \chi_n^{(\ell)})$ 
6:    $\text{hash}_n^{(\ell)} \leftarrow \text{HASH\_UPDATE}(\text{hash}_n^{(\ell-1)}, \text{inter\_hash}_n^{(\ell)})$ 
7:   ...
8:    $\ell \leftarrow \ell + 1$ 
9: end while
10:
11: procedure HASH_INTERSECTION( $o, m, s$ ) ▷ Pair object and primitive
12:    $h_1 \leftarrow \text{CANTOR\_PAIR}(o, m)$  ▷ Pair with interaction type
13:    $h_2 \leftarrow \text{CANTOR\_PAIR}(h_1, s)$ 
14:   return  $h_2$ 
15: end procedure
16:
17: procedure HASH_UPDATE( $\text{curr\_hash}, \text{inter\_hash}$ ) ▷ Update rolling hash
18:   return  $(1373 \cdot \text{curr\_hash} + \text{inter\_hash})$ 
19: end procedure
20:
21: procedure CANTOR_PAIR( $x, y$ )
22:    $z \leftarrow \frac{x^2+x+2xy+3y+y^2}{2}$ 
23:   return  $z$ 
24: end procedure

```

Cantor pairing function is a mapping between pairs of natural numbers and single natural numbers [15]. This bijective function guarantees that distinct input pairs always yield distinct output values, making it suitable for combining multiple values into a single hash. Hashing of the interaction first pairs the object and primitive indices, then combines the resulting value with the interaction type.

The `HASH_UPDATE` function updates the sample hash value $\text{hash}_n^{(\ell)}$ through a polynomial rolling hash function [16] with a prime base of 1373. For each new intersection, it updates the running hash by multiplying the current hash value by the prime number and adding the new intersection hash. When a specular chain candidate is considered for a target $k \in \{1, \dots, N_T\}$, the hash value of the candidate is paired with the target index k to form a new hash value

$$\text{hash}_{n,k}^{(\ell)} = \text{CANTOR_PAIR}(\text{hash}_n^{(\ell)}, k). \quad (20)$$

This ensures that the specular chain candidate is unique for each target.

Practical Considerations

To track information about previously encountered specular chains, an array of integers of size N_H is allocated in memory and initialized to zero, referred to as the *hash array*. When considering a specular chain candidate for target k with corresponding hash value $\text{hash}_{n,k}^{(\ell)}$, the corresponding index $i_{n,k}^{(\ell)}$ in the hash array is computed as

$$i_{n,k}^{(\ell)} = \text{hash}_{n,k}^{(\ell)} \mod N_H. \quad (21)$$

If the hash array entry at index $i_{n,k}^{(\ell)}$ is greater than zero, it indicates that this path has already been encountered and should be discarded. Otherwise, the entry is incremented by one and the candidate is retained for further processing by the image method. It is important to note that the index $i_{n,k}^{(\ell)}$ is not guaranteed to be unique for different specular chains. When two different specular chains share the same index value despite being different, the path solver will discard one of them. Such an event is referred to as a *collision*, and can result in

the discarding of valid paths. However, when two candidates are identical, their indices values will also be identical.

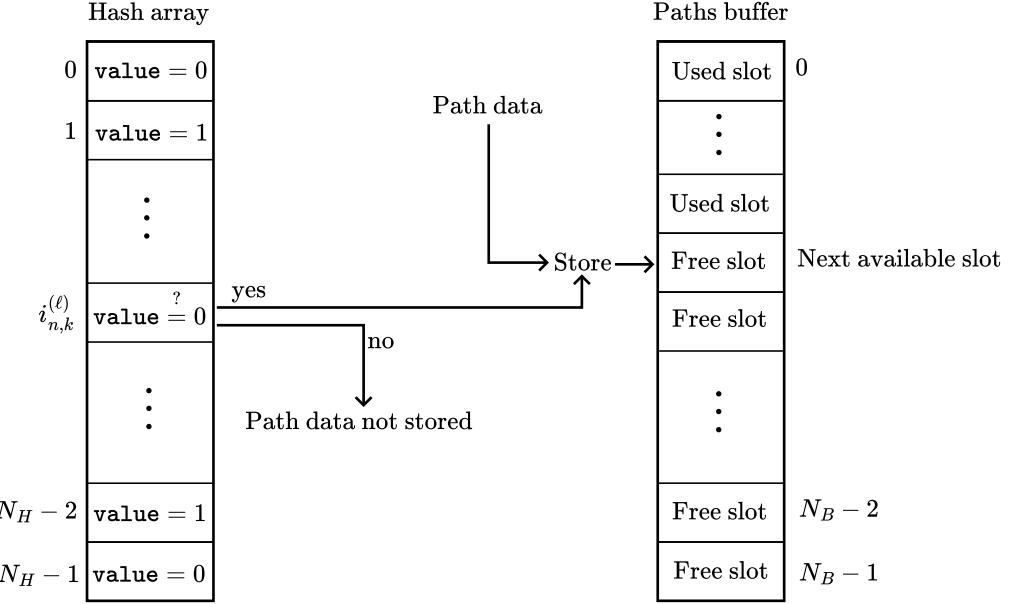


Figure 15: The hash array is used to track encountered path candidates. New candidates are stored in the buffer if they haven't been encountered before. Non-specular chains are added to the buffer without a uniqueness check (not depicted in the figure).

Valid paths and candidates are stored in a buffer of size N_B . This buffer holds, for each path, the sequence of interaction types, vertices, intersected objects, primitives, and more. When a valid path is found or a candidate is identified as unique, indicated by its corresponding entry $i_{n,k}^{(\ell)}$ (21) in the hash array being set to zero, the path data are stored in the buffer. It is important to note that the index $i_{n,k}^{(\ell)}$ is solely used for indexing the hash array and not for storing path data in the buffer. Instead, path data are stored contiguously in the next available slot of the path buffer, as shown in Figure 15. If the buffer becomes full, any newly found paths or candidates are discarded. Discarding paths rather than overwriting existing ones is based on the reason that paths with lower depth, which are added to the buffer first as the SBR loop advances from $\ell = 0$ to $\ell = L$, typically carry more energy and should not be replaced by higher depth paths when the buffer becomes full. Consequently, the buffer size N_B determines the maximum number of paths and candidates that can be stored.

Due to the substantial amount of data stored for each path, setting N_B to excessively high values can lead to memory issues. This necessitates separating the hash array size N_H from the buffer size N_B . This is because a small hash array size can increase the probability of different hash values mapping to the same array index through the modulo operation (21), leading to collisions. To reduce the risk of collisions, N_H is set to

$$N_H = \max(N_B, 10^6) \quad (22)$$

ensuring a minimum array size of 10^6 elements. In the interface of the Sionna RT built-in path solver, N_B corresponds to the `samples_per_src` parameter, and N_H corresponds to `max_num_paths_per_src`.

3.1.4. Potential Improvements

Several potential improvements can be made to enhance the SBR-based candidate generator. To begin with, optimizing the sampling of ray directions can be achieved by incorporating importance sampling based on the transmit antenna pattern for initial ray spawning and the scattering pattern for diffusely reflected rays. This approach is motivated by the substantial sample efficiency gains shown in Section 3.1.2 through the use of importance sampling. By applying optimized sampling strategies for ray directions, further improvements may be realized as illustrated in Figure 16.

Moreover, the current method for handling multiple targets involves iterating over all targets during each

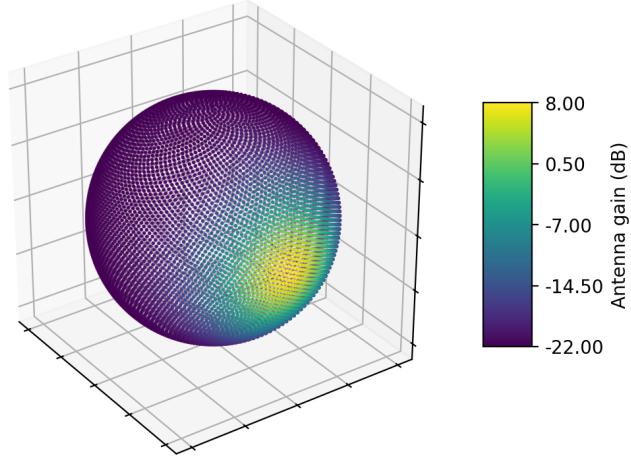


Figure 16: Gain of the 3GPP TR 38.901 [17, Table 7.3-1] antenna pattern for each sample direction on a Fibonacci lattice with 10^4 points. The majority of sample directions exhibit low gain, indicating that importance sampling of ray directions may be beneficial.

SBR loop iteration. This approach may be optimized by selecting only targets near intersection points or the source, which will enhance scalability when dealing with a large number of targets. Figure 17b illustrates this, showing the compute time of the candidate generator and the average number of candidates per target found as the number of targets varies. In this experiment, the built-in Sionna RT scene “simple street canyon” depicted in Figure 17a, was used. Targets were sampled uniformly at random on a plane parallel to the ground at an elevation of 1.5 meters. The maximum depth L was set to 5, diffuse reflections were disabled, and the number of samples N_S was set to 10^6 . As observed, the compute time increases with the number of targets due to the iteration over all targets in each **SBR** loop iteration. However, the number of candidates per target remains stable, although with variations due to hash collisions. Note that, as discussed in Section 3.2, most candidates do not result in valid paths.

Scaling with the number of sources presents its own set of challenges. When N_S samples are generated per source, the total number of samples produced by the candidate generator becomes $N_S \cdot N_O$, where N_O denotes the number of sources. This results in increased computation time as the number of sources grows, as illustrated in Figure 17c. However, the number of candidates per source remains stable. In this experiment, a single target is positioned at the center of the scene at an elevation of 1.5 meters, while sources are randomly sampled on a plane parallel to the ground at an elevation of 70 meters, above all buildings. Recall that variations in the number of candidates per source may occur due to collisions. For a comprehensive view, Figure 17d presents results where the total number of samples is kept constant at 10^6 , resulting in $N_S = \left\lfloor \frac{10^6}{N_O} \right\rfloor$ samples per source. In this scenario, the computation time increases at a much slower rate with the number of sources, but the number of candidates found per source decreases rapidly as the number of sources increases.

3.2. Image Method-based Candidate Processing

The image method processes path candidates to determine valid paths that are specular chains, or end with specular suffixes. Consider a candidate path

$$c_n^{(\ell)} = \left(\mathbf{s}, \left(o_n^{(1)}, m_n^{(1)}, \chi_n^{(1)} \right), \dots, \underbrace{\left(o_n^{(\ell_d+1)}, m_n^{(\ell_d+1)}, \chi_n^{(\ell_d+1)} \right), \dots, \left(o_n^{(\ell)}, m_n^{(\ell)}, \chi_n^{(\ell)} \right)}_{\text{Specular suffix}}, \mathbf{t} \right) \quad (23)$$

where ℓ denotes the path depth, ℓ_d is the specular suffix index defined as in (10), and \mathbf{t} is the target of the path. Recall that the sequence of interactions $\chi^{(\ell)} = (\chi_n^{(1)}, \dots, \chi_n^{(\ell)})$ was determined by the **SBR**-based candidate

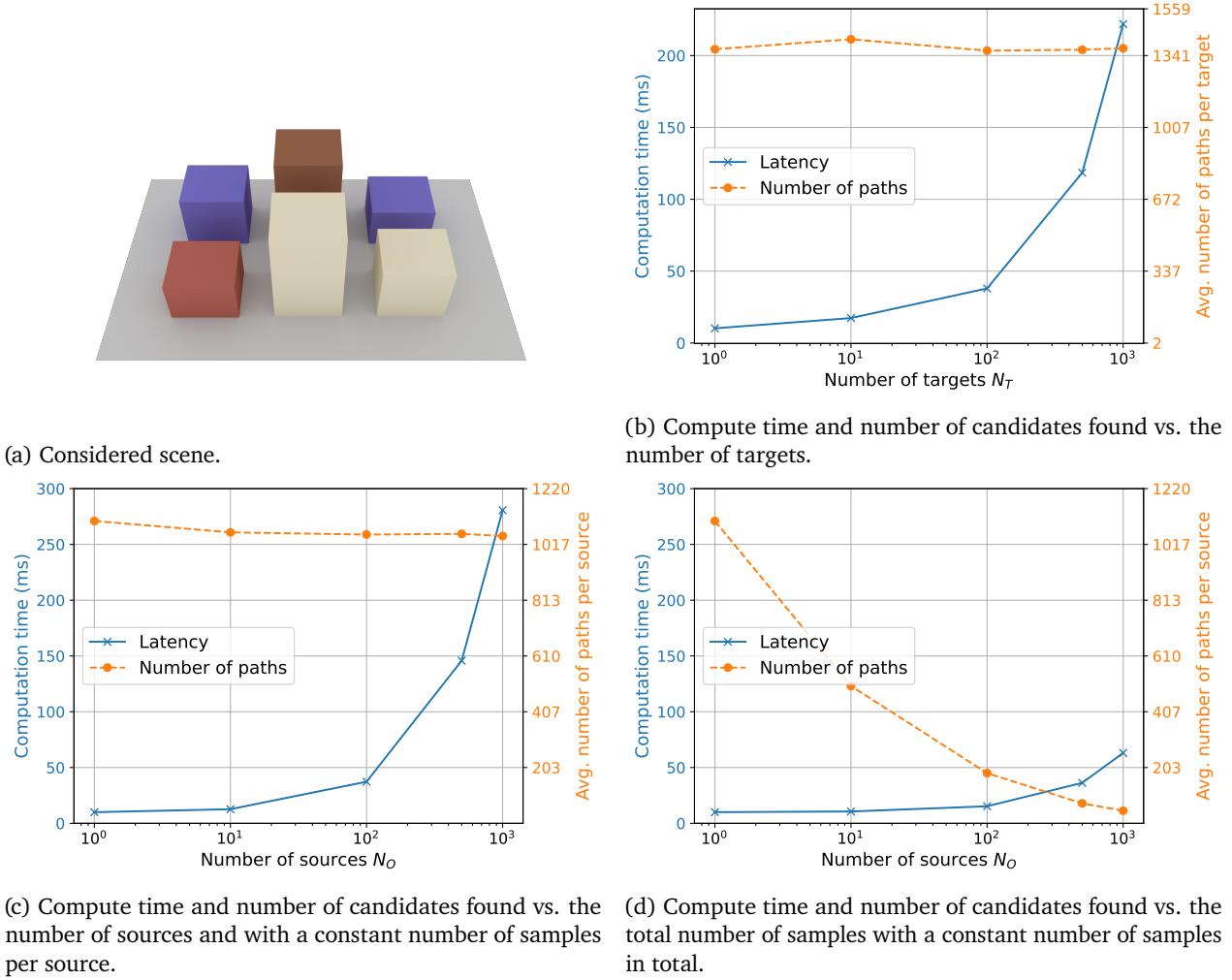


Figure 17: Computation time and number of candidates found as a function of the number of sources and targets. The experiments are conducted on an NVIDIA RTX 4090 GPU.

generator. For path candidates that end with a specular suffix but are not specular chains (i.e. $\ell_d > 0$), the image method processes only the specular suffix. In these cases, the vertex of the last diffuse reflection $v_n^{(\ell_d)}$ serves as the source point for the image method calculations. Since the processing algorithm is identical for both specular chains and specular suffixes, we treat them equivalently in the following discussion. For simplicity of notation, we will refer to both types as specular chains and assume $\ell_d = 0$ throughout the remainder of this section.

Figure 18 illustrates the image method process, which consists of two main steps. First, the method iteratively computes images of the source by reflecting it across each plane at which a specular reflection occurs. Since refracted paths are modeled without angular deflection, incorporating refraction into the image method is straightforward: refraction events may simply be skipped when computing the source images, as they do not alter the ray's direction. Then, the algorithm backtracks from the target to determine the actual path vertices by intersecting lines between successive image sources.

Formally, the image method iteratively computes images of the source $\tilde{s}_n^{(i)}$ for $i = 0, \dots, \ell$, starting from the original source s :

$$\tilde{s}_n^{(i)} = \begin{cases} s & \text{if } i = 0 \\ \tilde{s}_n^{(i-1)} - 2 \left(\tilde{s}_n^{(i-1)} - p_n^{(i)} \right)^T \hat{n}_n^{(i)} & \text{if } \chi_n^{(i)} = \mathcal{R} \\ \tilde{s}_n^{(i-1)} & \text{if } \chi_n^{(i)} = \mathcal{T} \end{cases} \quad (24)$$

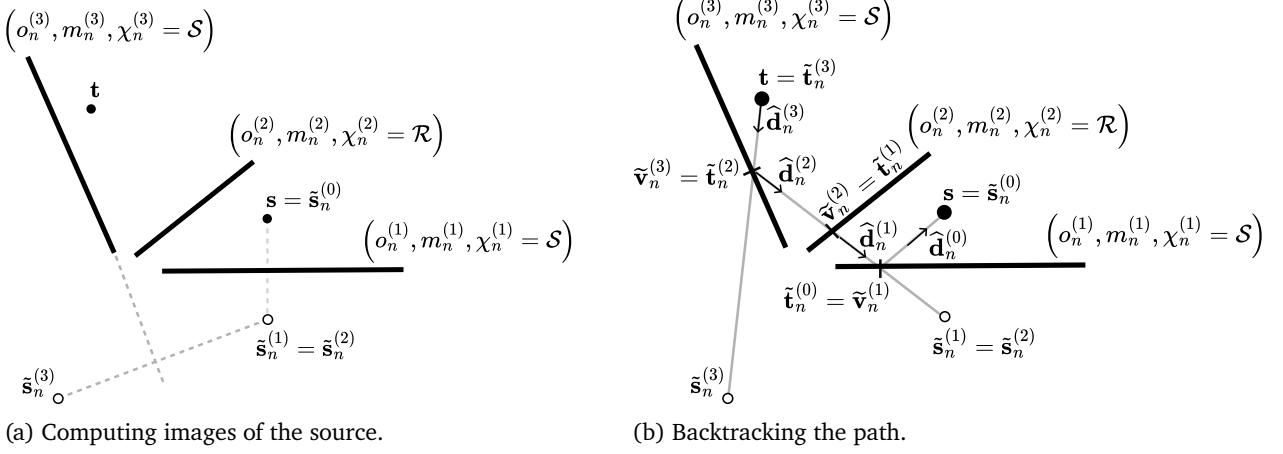


Figure 18: Two-step process of the image method: (a) source image computation and (b) path backtracking. Shown in 2D for clarity.

Here, $\mathbf{p}_n^{(i)}$ represents any point on the plane containing the primitive $(o_n^{(i)}, m_n^{(i)})$, and $\hat{\mathbf{n}}_n^{(i)}$ denotes the surface's normal vector. For specular reflections ($\chi_n^{(i)} = \mathcal{R}$), the image is computed by reflecting the previous image across the surface plane, while for refractions ($\chi_n^{(i)} = \mathcal{T}$), the image position remains unchanged.

During backtracking, the algorithm traces rays iteratively from the target \mathbf{t} back to the source \mathbf{s} by intersecting lines between successive image points. Specifically, a sequence of ℓ rays $(\tilde{\mathbf{t}}_n^{(i)}, \tilde{\mathbf{d}}_n^{(i)})$, $i = \ell, \dots, 0$, is traced according to

$$\begin{aligned} \tilde{\mathbf{t}}_n^{(i)} &= \begin{cases} \mathbf{t} & \text{if } i = \ell \\ \tilde{\mathbf{v}}_n^{(i+1)} & \text{if } 1 \leq i < \ell \end{cases} \\ \tilde{\mathbf{d}}_n^{(i)} &= \frac{\tilde{\mathbf{s}}_n^{(i)} - \tilde{\mathbf{t}}_n^{(i)}}{\|\tilde{\mathbf{s}}_n^{(i)} - \tilde{\mathbf{t}}_n^{(i)}\|_2} \\ \tilde{\mathbf{v}}_n^{(i)} &= \text{intersection}\left(\left(\tilde{\mathbf{t}}_n^{(i)}, \tilde{\mathbf{d}}_n^{(i)}\right), \left(o_n^{(i)}, m_n^{(i)}\right)\right), \text{only if } i > 0 \end{aligned} \quad (25)$$

where $\tilde{\mathbf{v}}_n^{(i)}$ is the intersection point between the ray $(\tilde{\mathbf{t}}_n^{(i)}, \tilde{\mathbf{d}}_n^{(i)})$ and the primitive $(o_n^{(i)}, m_n^{(i)})$, yielding the path vertex $\tilde{\mathbf{v}}_n^{(i)}$ as shown in Figure 18b. The path is considered valid only if two conditions are met: (i) each ray successfully intersects its corresponding primitive $(o_n^{(i)}, m_n^{(i)})$, and (ii) the line segment between $\tilde{\mathbf{t}}_n^{(i)}$ and $\tilde{\mathbf{v}}_n^{(i)}$ is unobstructed by any other scene geometry. If either condition fails, the path will be discarded. The case $i = 0$ corresponds to the final segment of the path, which connects to the source \mathbf{s} . If the path is valid, then the returned path will be (14).

Typically, most of the candidates are discarded by the image method. This is illustrated in Figure 19, which shows the number of candidates discarded as a function of the path depth for a single source and target. The considered scene is shown in Figure 19a. The number of samples N_S is 10^6 and diffuse reflection is disabled. As seen in Figure 19b, most of the candidates are discarded by the image method because only specular chains are generated by the candidate generator, and most of them do not result in valid paths.

3.3. Channel Coefficients, Delays, and Doppler Shifts Computation

The final step of the path solver computes complex-valued channel coefficients and real-valued delays for each valid path (see Figure 9). For clarity, we will focus our discussion on a single valid path with index n and depth ℓ , though the same procedure applies to all valid paths computed by both the SBR-based candidate generator and image method. We denote the path vertices as $\mathbf{v}_n^{(i)}$, $i = 1, \dots, \ell$, which may correspond to vertices previously computed by the image method (denoted earlier as $\tilde{\mathbf{v}}_n^{(i)}$). The directions of propagation

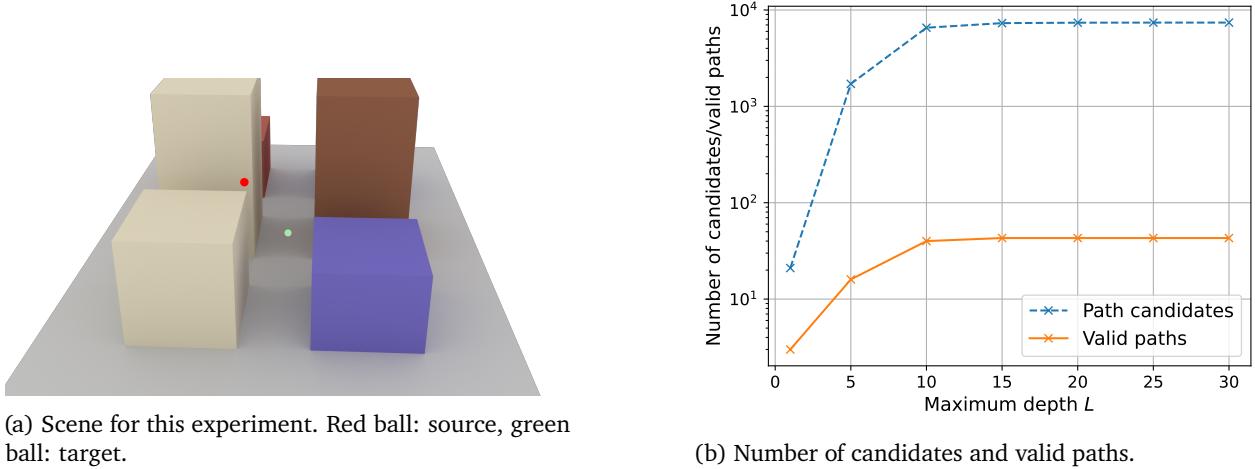


Figure 19: The image method typically discards most candidates.

between vertices are represented by unit vectors $\hat{\mathbf{k}}_n^{(i)}$, $i = 0, \dots, \ell$, defined as

$$\hat{\mathbf{k}}_n^{(i)} = \frac{\mathbf{v}_n^{(i+1)} - \mathbf{v}_n^{(i)}}{\|\mathbf{v}_n^{(i+1)} - \mathbf{v}_n^{(i)}\|_2}, \quad i = 0, \dots, \ell \quad (26)$$

where $\mathbf{v}_n^{(0)} = \mathbf{s}$ is the source and $\mathbf{v}_n^{(\ell+1)} = \mathbf{t}$ is the target. Note that these propagation directions may differ from those used during the initial SBR loop described in Section 3.1 due to the refinement by the image method.

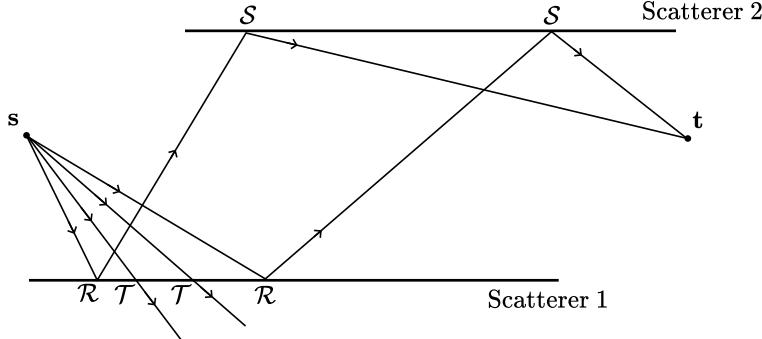


Figure 20: Due to the random sampling of interaction types, only some of the ray tubes that should reach the diffusely reflective surface (Scatterer 2) are traced. In this example, two out of four initial rays are transmitted when interacting with Scatterer 1, while the other two are reflected and reach the diffusely reflective surface.

Ensuring Energy Conservation

As explained in Section 3.1.2, at each interaction point, an interaction type is randomly chosen based on the probability distribution \mathcal{Q} (17). Initially, the energy emitted by the source is distributed across N_S samples according to the transmit antenna pattern (see Section A.3). Since only one interaction type is selected at each interaction point, ray tubes that do not match the chosen interaction type are discarded, as illustrated in Figure 20. The remaining ray tubes must therefore compensate for the discarded ones to maintain energy conservation. Specular reflection and refraction on planar surfaces only change the direction of the ray tube's propagation. Given that there is a finite number of specular chains with a specific depth, the channel gain (5) observed at the target due to specular chains is accurate if all specular chains are identified by the path solver, which is assumed to be the case. This assumption will be reasonable if the number of samples N_S is sufficiently large.

In the context of diffuse reflections, an incident ray tube scatters an amount of energy that is determined by its footprint on the diffusely reflective surface (see Section A.8). Due to the random sampling of interaction

types, only a portion of the ray tubes that should reach a diffusely reflective surface are actually traced (see Figure 20), which requires compensating for the untraced ones. For a given path $p^{(\ell)}$ that includes a diffuse reflection and has a specular suffix index ℓ_d (10), we denote by $\Pr(\chi_n^{(\ell_d)}) = \prod_{\ell'=1}^{\ell_d} q(\chi_n^{(\ell')})$ the probability of sampling the prefix leading to the last diffuse reflection point, where $q(\cdot)$ is defined in (17). Consequently, on average, only a fraction $\Pr(\chi_n^{(\ell_d)})$ of the ray tubes that should reach the diffusely reflective surface are actually traced. To account for this, the electric field is scaled by $1/\sqrt{\Pr(\chi_n^{(\ell_d)})}$ at the point of diffuse reflection.

Algorithm 2 Electric field computation

```

1:  $\mathbf{E}_n^{(0)} \leftarrow \text{TX\_PATTERN}(\hat{\mathbf{k}}_n^{(0)})$                                 ▷ Electric field vector
2:  $\tau_n^{(0)} \leftarrow 0$                                                                ▷ Cumulative delay
3:  $r_n^{(0)} \leftarrow 0$                                                                ▷ Cumulative ray tube length
4:  $\gamma_n^{(0)} \leftarrow 1$                                                                ▷ Cumulative path probability
5:  $\ell' \leftarrow 1$ 

6: while not <Stop condition> do
7:    $\mathbf{T}_n^{(\ell')} \leftarrow \text{EVALUATE\_MATERIAL}(o_n^{(\ell')}, \chi_n^{(\ell')}, \mathbf{E}_n^{(\ell'-1)}, \hat{\mathbf{k}}_n^{(\ell'-1)}, \hat{\mathbf{k}}_n^{(\ell')})$ 
8:    $\mathbf{E}_n^{(\ell')} \leftarrow \mathbf{T}_n^{(\ell')} \mathbf{E}_n^{(\ell'-1)}$ 
9:    $\gamma_n^{(\ell')} \leftarrow \gamma_n^{(\ell'-1)} \cdot q_n^{(\ell')} (\chi_n^{(\ell')})$ 
10:  if  $\chi_n^{(\ell')} = \mathcal{S}$  then
11:     $\mathbf{E}_n^{(\ell')} \leftarrow \frac{\mathbf{E}_n^{(\ell')}}{\sqrt{\gamma_n^{(\ell')}}}$ 
12:     $\gamma_n^{(\ell')} \leftarrow 1$ 
13:     $r_n^{(\ell'-1)} \leftarrow 0$ 
14:  end if
15:   $r_n^{(\ell')} \leftarrow r_n^{(\ell'-1)} + \|\mathbf{v}_n^{(\ell')} - \mathbf{v}_n^{(\ell'-1)}\|_2$ 
16:   $\tau_n^{(\ell')} \leftarrow \tau_n^{(\ell'-1)} + \frac{\|\mathbf{v}_n^{(\ell')} - \mathbf{v}_n^{(\ell'-1)}\|_2}{c}$ 
17:   $\ell' \leftarrow \ell' + 1$ 
18: end while
19:  $\tau_n \leftarrow \tau_n^{(\ell)} + \frac{\|\mathbf{t} - \mathbf{v}_n^{(\ell)}\|_2}{c}$ 
20: if  $\chi_n^{(\ell)} = \mathcal{S}$  then
21:    $r_n^{(\ell-1)} \leftarrow 0$ 
22: end if
23:  $r_n^{(\ell)} \leftarrow r_n^{(\ell-1)} + \|\mathbf{t} - \mathbf{v}_n^{(\ell)}\|_2$ 
24:  $\mathbf{E}_n^{(r)} \leftarrow \text{RX\_PATTERN}(-\hat{\mathbf{k}}_n^{(\ell)})$ 
25:  $a_n \leftarrow \frac{\lambda}{4\pi r_n^{(\ell)}} \left( (\mathbf{E}_n^{(\ell)})^\top \mathbf{E}_n^{(r)} \right)$ 
26: return  $a_n, \tau_n$ 

```

Path Coefficient and Delay Computation

Algorithm 2 details the computation of the electric field propagation along a valid path, leveraging the EM background presented in Appendix A. The algorithm begins by initializing the electric field vector $\mathbf{E}_n^{(0)}$ using the transmit antenna's pattern evaluated in the initial propagation direction $\hat{\mathbf{k}}_n^{(0)}$. Note that the path solver supports dual polarization by considering two independent electric field vectors for each traced path. The variable $\tau_n^{(0)}$ tracks the cumulative path delay and is initialized to zero, and the variable $r_n^{(0)}$ tracks the ray tube length and is also initialized to zero. The variable $\gamma_n^{(0)}$ is used to track the cumulative path probability and is initialized to one (see previous paragraph). For each interaction point along the path, the electric field is updated by a transformation matrix $\mathbf{T}_n^{(\ell')}$, which is computed based on the intersected object $o_n^{(\ell')}$, interaction type $\chi_n^{(\ell')}$, incident field $\mathbf{E}_n^{(\ell'-1)}$, and both incident $\hat{\mathbf{k}}_n^{(\ell'-1)}$ and scattered $\hat{\mathbf{k}}_n^{(\ell')}$ directions. The cumulative path probability

$\gamma_n^{(\ell')}$ is updated by multiplying it with the interaction probability $q_n^{(\ell')} \left(\chi_n^{(\ell')} \right)$. When a diffuse reflection occurs, the electric field is normalized by $\sqrt{\gamma_n^{(\ell')}}$ to maintain energy conservation and $\gamma_n^{(\ell')}$ is reset to one and the ray tube length $r_n^{(\ell')}$ to zero. The ray tube length $r_n^{(\ell')}$ is updated by adding the distance between the current and previous vertex, and the delay $\tau_n^{(\ell')}$ is updated by adding the time it takes to travel that distance, where c is the speed of light. After replaying all interactions, the final path segment to the target is incorporated into the delay and ray tube length. Finally, the receive antenna pattern is evaluated for the incoming direction, and the channel coefficient is computed following Section A.6.

Remark

The transformation matrices $\mathbf{T}_n^{(\ell')}$ and the electric field vectors $\mathbf{E}_n^{(\ell')}$ are complex-valued, but are implemented using their real-valued representations defined as

$$\begin{bmatrix} \Re(\mathbf{T}_n^{(\ell')}) & -\Im(\mathbf{T}_n^{(\ell')}) \\ \Im(\mathbf{T}_n^{(\ell')}) & \Re(\mathbf{T}_n^{(\ell')}) \end{bmatrix} \quad (27)$$

for matrices and

$$\begin{bmatrix} \Re(\mathbf{E}_n^{(\ell')}) \\ \Im(\mathbf{E}_n^{(\ell')}) \end{bmatrix} \quad (28)$$

for vectors, where $\Re(\cdot)$ and $\Im(\cdot)$ denote the real and imaginary parts, respectively.

3.3.1. Handling Multiple Antennas at the Transmitter and Receiver

As explained in the Introduction (Section 1), Sionna RT supports antenna arrays of any size at both the transmitter and receiver, utilizing either synthetic or non-synthetic arrays. In the case of non-synthetic arrays, each transmit antenna is treated as a source and each receive antenna as a target, meaning paths are traced from every transmit antenna, and the solver calculates paths for each transmit-receive antenna pair. For synthetic arrays, paths originate from a single “virtual” source at the center of the transmitter array, and a single “virtual” target is considered at the center of every receiver array. Channel coefficients are then computed by applying phase shifts synthetically. This method allows for efficient scaling with the number of antennas, but it is an approximation that is valid only when the transmitter and receiver array sizes are small relative to the distance from the radio devices to each other and to the scatterers. Formally, for synthetic arrays, the channel matrix for the n -th path is expressed as:

$$\mathbf{H}_{\text{array},n} = a_n \mathbf{u}_A^{\text{rx}} (\mathbf{u}_A^{\text{tx}})^T \quad (29)$$

where a_n is the channel coefficient for the n -th path, calculated using Algorithm 2. The array response vectors for the transmit (\mathbf{u}_A^{tx}) and receive (\mathbf{u}_A^{rx}) antenna arrays are defined as:

$$\begin{aligned} \mathbf{u}_A^{\text{tx}} &= \left[e^{j \frac{2\pi}{\lambda} (\hat{\mathbf{k}}_n^{(0)})^T \mathbf{d}_1^{\text{tx}}}, \dots, e^{j \frac{2\pi}{\lambda} (\hat{\mathbf{k}}_n^{(0)})^T \mathbf{d}_{N_A^{\text{tx}}}^{\text{tx}}} \right]^T, \\ \mathbf{u}_A^{\text{rx}} &= \left[e^{j \frac{2\pi}{\lambda} (-\hat{\mathbf{k}}_n^{(\ell)})^T \mathbf{d}_1^{\text{rx}}}, \dots, e^{j \frac{2\pi}{\lambda} (-\hat{\mathbf{k}}_n^{(\ell)})^T \mathbf{d}_{N_A^{\text{rx}}}^{\text{rx}}} \right]^T. \end{aligned} \quad (30)$$

Here, $\hat{\mathbf{k}}_n^{(0)}$ ($\hat{\mathbf{k}}_n^{(\ell)}$) represents the departure (arrival) direction of the n -th path, N_A^{tx} (N_A^{rx}) denotes the number of antennas in the transmit (receive) array, and \mathbf{d}_i^{tx} (\mathbf{d}_i^{rx}) indicates the relative position of the i -th antenna element to the virtual source (target) in the transmit (receive) array.

3.3.2. Doppler Shifts

The impact of moving scene objects on the paths can be simulated in two ways. The first approach involves moving objects in small incremental steps along their trajectories and recomputing the propagation paths at each step. While this method provides the highest accuracy, it is computationally expensive. However, when

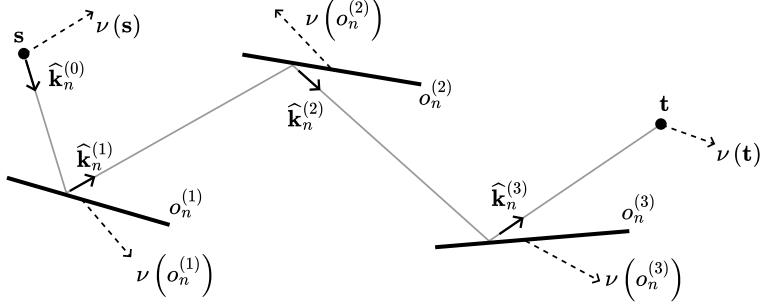


Figure 21: Each object as well as the source and target has an associated velocity vector that contributes to the total Doppler shift. Shown in 2D for clarity.

trajectory lengths are small (not exceeding a few wavelengths), a more efficient approach may be used. In this alternative method, we assign a velocity vector $v(o) \in \mathbb{R}^3$ to each object o and compute the time evolution by accumulating Doppler shifts along the propagation paths.

Figure 21 illustrates this approach. When replaying the valid paths using Algorithm 2, the Doppler shift for each path n is computed by accumulating the contributions from individual objects:

$$\nu_n^{(\ell')} = \begin{cases} 0 & \text{if } \ell' = 0 \\ \nu_n^{(\ell'-1)} + \frac{v(o_n^{(\ell')})^\top (\hat{k}_n^{(\ell')} - \hat{k}_n^{(\ell'-1)})}{\lambda} & \text{if } 1 \leq \ell' < \ell \end{cases} \quad (31)$$

where λ is the wavelength. This results in the total Doppler shift $\nu_n^{(\ell)}$ caused by object movements along the path. The final Doppler shift observed at the target combines the accumulated shift from moving objects with the shifts caused by source and target motion:

$$\nu_n = \nu_n^{(\ell)} + \frac{v(s)^\top \hat{k}_n^{(0)}}{\lambda} - \frac{v(t)^\top \hat{k}_n^{(\ell)}}{\lambda}. \quad (32)$$

where $v(s) \in \mathbb{R}^3$ and $v(t) \in \mathbb{R}^3$ denote the velocity vectors of the source and target, respectively.

4. Radio Map Solver

In the previous sections, we have detailed the path solver, which computes propagation paths and the related electric fields from a source to a target. Now, we turn our attention to the radio map solver, which computes a *radio map* (also known as *coverage map* or *power map*) for a given scene and source. While we describe the algorithm for a single source, Sionna RT can compute radio maps for multiple sources in parallel.

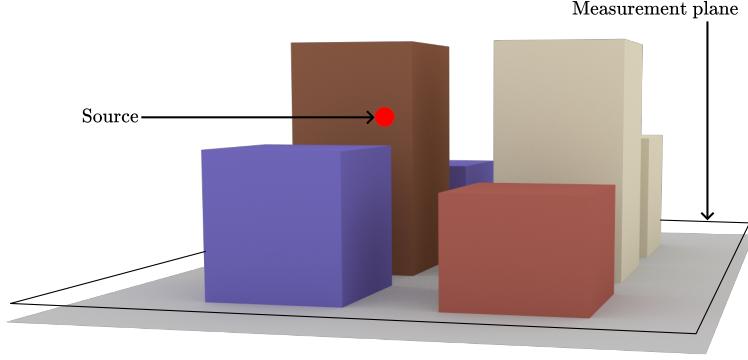
A radio map estimates the channel gain (5) from a source to each point on a *measurement plane*, as illustrated in Figure 22. The measurement plane is partitioned into *measurement cells* of identical size. For each cell, the solver estimates the average channel gain that would be observed by a target placed within that cell. Note that the measurement plane does not interact with the electromagnetic waves, but only serves to capture the paths that intersect it.

In Section 4.1, we present a definition of radio maps that allows for their efficient computation using Monte Carlo integration. This approach, detailed in Section 4.2, makes the computation of radio maps both fast and scalable.

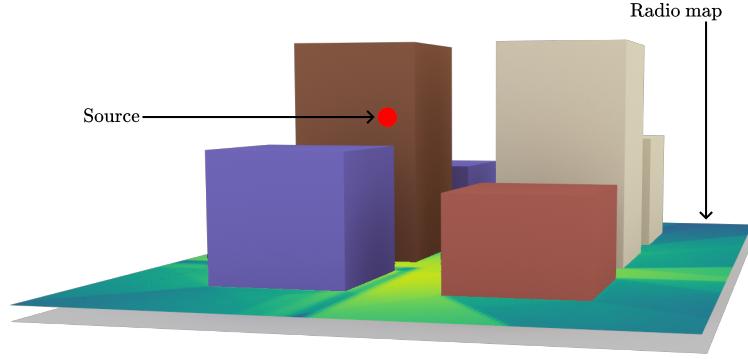
4.1. Definition of a Radio Map

Consider a point source s . The measurement plane, denoted as \mathcal{M} , is partitioned into N_M cells, each with an equal area $|C|$. The radio map value for the i -th cell, where $i \in \{1, \dots, N_M\}$, is defined as:

$$C_i = \frac{1}{|C|} \int_{C_i} \bar{e}(t) dt. \quad (33)$$



(a) Scene with a measurement plane.



(b) Radio map for the measurement plane in (a).

Figure 22: (a) A measurement plane is placed in the scene to capture paths without affecting their propagation, (b) the resulting radio map showing the spatial distribution of channel gains.

Here, $\bar{e}(t)$ denotes the channel gain at the position t on the measurement plane, considering all paths originating from the source s . Assuming a discrete set of paths impacts the measurement plane at t , $\bar{e}(t)$ can be expressed as

$$\bar{e}(t) = \sum_{\ell=0}^{\infty} \sum_{p \in \mathcal{P}(s, \ell, t)} e(p) \quad (34)$$

where $\mathcal{P}(s, \ell, t)$ represents the set of all paths with depth ℓ that originate from s and intersect the measurement plane at t . The term $e(p)$ is defined as:

$$e(p) = \left\| \frac{\lambda}{4\pi} \left(\prod_{\ell'=1}^{\ell} \mathbf{T}^{(\ell')} \right) \mathbf{C}_T(\theta_T, \varphi_T) \right\|_2^2. \quad (35)$$

In this context, λ is the wavelength, ℓ is the number of scatterers along the path p , \mathbf{C}_T is a complex-valued vector of dimension 2 representing the transmit antenna pattern evaluated at the angles of departure of p , (θ_T, φ_T) , and $\mathbf{T}^{(\ell')}$ is a 2×2 complex-valued matrix modeling the interaction with the ℓ' -th scatterer. Unlike in (4), the receive antenna pattern is not applied here. Instead, the squared norm of the electric field is used. This is equivalent to assuming that a receiver positioned on the measurement plane uses a dual-polarized isotropic antenna, and that both components are combined non-coherently.

While the definition (33) is comprehensive, it is not directly practical for computation. A straightforward but inefficient method to compute a radio map is to calculate the paths for each measurement cell using the path solver (Section 3) by placing one or more targets within each cell. However, this approach becomes computationally prohibitive as the number of cells increases. Therefore, we reformulate the radio map definition (33) to enable efficient computation via Monte Carlo integration.

To build intuition, let's first consider the scenario where all paths from the source s to the measurement cells are specular chains, meaning they contain no diffuse reflections. In this case, for a given scene and

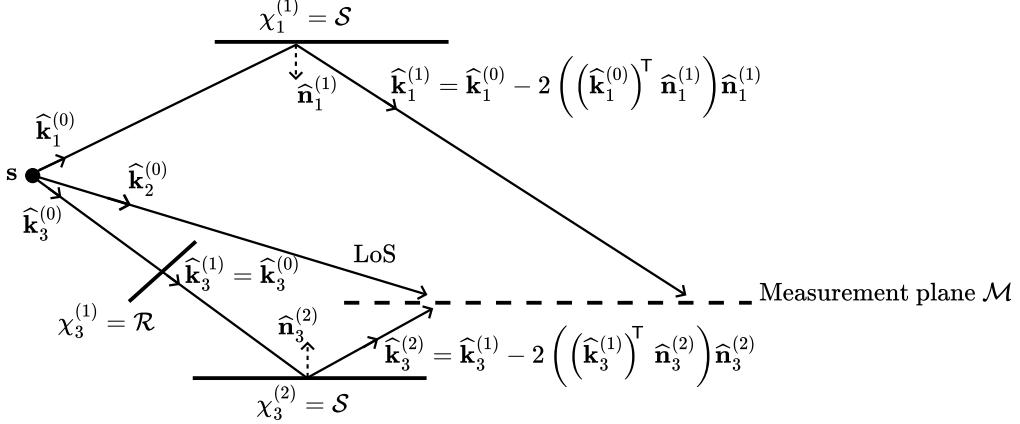


Figure 23: Three specular chains, each indexed by a subscript. The sequence of interaction types and the departure direction uniquely determine the directions of the scattered rays, and therefore the paths. Shown in 2D for clarity.

source s , a path of depth ℓ is fully characterized by its sequence of interaction types $\chi^{(\ell)} = (\chi^{(1)}, \dots, \chi^{(\ell)})$ and its departure direction $\hat{\mathbf{k}}^{(0)}$. This is because, in specular chains, the direction of the scattered wave at each interaction point is uniquely determined by the interaction type and the direction of incidence, as illustrated in Figure 23. In this case, the definition (33) can be reformulated over the unit sphere Ω centered on s , which represents all possible ray departure directions. To achieve this, we define $p(s, \hat{\mathbf{k}}^{(0)}, \chi^{(\ell)}, t)$ as the path of depth ℓ from s to $t \in \mathcal{M}$, characterized by the departure direction $\hat{\mathbf{k}}^{(0)}$ and the sequence of interaction types $\chi^{(\ell)}$. The length of the corresponding ray tube is denoted by r , and the angle between the plane normal and the path's incident direction is denoted by θ . By applying the change of variables $dt = \frac{r^2}{\cos \theta} d\hat{\mathbf{k}}^{(0)}$, we can reformulate the definition (33) as:

$$C_i = \frac{1}{|C|} \sum_{\ell=0}^{\infty} \sum_{\chi^{(\ell)} \in \{\mathcal{R}, \mathcal{T}\}^\ell} \int_{\Omega} e(p(s, \hat{\mathbf{k}}^{(0)}, \chi^{(\ell)}, t)) \mathbb{I}(t \in C_i) \frac{r^2}{\cos \theta} d\hat{\mathbf{k}}^{(0)}. \quad (36)$$

In this context, t denotes the position on the measurement plane where the path intersects. The double summation is over all path depths $\ell = 0, 1, \dots$ (with $\ell = 0$ corresponding to LoS) and all possible sequences of interaction types $\chi^{(\ell)}$, excluding diffuse reflections. The indicator function $\mathbb{I}(t \in C_i)$ equals 1 if t is within C_i and 0 otherwise. If the path does not intersect the measurement plane, then $\mathbb{I}(t \in C_i) = 0$. The integration is carried out over the unit sphere, where $d\hat{\mathbf{k}}^{(0)}$ represents the infinitesimal solid angle element corresponding to the ray tube in the direction $\hat{\mathbf{k}}^{(0)}$.

To extend the definition to include diffuse reflections, we first modify the definition (36) to explicitly include the sequence of propagation directions of the scattered rays at the interaction points, denoted as $\bar{\mathbf{k}}^{(\ell)} = (\hat{\mathbf{k}}^{(1)}, \dots, \hat{\mathbf{k}}^{(\ell)})$. For this purpose, we introduce $\bar{\mathbf{n}}^{(\ell)} = (\hat{\mathbf{n}}^{(1)}, \dots, \hat{\mathbf{n}}^{(\ell)})$, representing the sequence of normals at these interaction points. We then define the function $\bar{f}(\cdot)$, which ensures that the direction of the scattered rays aligns with the interaction type, as follows:

$$\bar{f}(\hat{\mathbf{k}}^{(0)}, \bar{\mathbf{k}}^{(\ell)}, \chi^{(\ell)}, \bar{\mathbf{n}}^{(\ell)}) = \prod_{\ell'=1}^{\ell} f(\chi^{(\ell')}, \hat{\mathbf{k}}^{(\ell'-1)}, \hat{\mathbf{k}}^{(\ell')}, \hat{\mathbf{n}}^{(\ell')}) \quad (37)$$

where

$$f(\chi, \hat{\mathbf{k}}_i, \hat{\mathbf{k}}_s, \hat{\mathbf{n}}) = \begin{cases} \delta(\hat{\mathbf{k}}_s - (\hat{\mathbf{k}}_i - 2(\hat{\mathbf{n}}^\top \hat{\mathbf{k}}_i) \hat{\mathbf{n}})) & \text{if } \chi = \mathcal{R} \\ \delta(\hat{\mathbf{k}}_s - \hat{\mathbf{k}}_i) & \text{if } \chi = \mathcal{T} \end{cases} \quad (38)$$

and, $\delta(\cdot)$ is the Dirac delta distribution. The definition in (36) can be then be reformulated as

$$C_i = \frac{1}{|C|} \sum_{\ell=0}^{\infty} \sum_{\chi^{(\ell)} \in \{\mathcal{R}, \mathcal{T}\}^\ell} \int_{\Omega} \int_{\Omega^\ell} e(p(s, \hat{k}^{(0)}, \bar{k}^{(\ell)}, \chi^{(\ell)}, t)) \mathbb{I}(t \in C_i) \bar{f}(\hat{k}^{(0)}, \bar{k}^{(\ell)}, \chi^{(\ell)}, \bar{n}^{(\ell)}) \frac{r^2}{\cos \theta} d\bar{k}^{(\ell)} d\hat{k}^{(0)} \quad (39)$$

Compared to (36), the definition in (39) integrates over all propagation directions of the scattered rays at the interaction points. The integral over Ω^ℓ combined with the function $\bar{f}(\cdot)$ merely filters out invalid directions and, strictly speaking, does not perform any other function. However, while the definition in (39) is equivalent to (36), it facilitates the extension to paths with diffuse reflections:

$$C_i = \frac{1}{|C|} \sum_{\ell=0}^{\infty} \sum_{\chi^{(\ell)} \in \mathcal{I}^\ell} \int_{\Omega} \int_{\Omega^\ell} e(p(s, \hat{k}^{(0)}, \bar{k}^{(\ell)}, \chi^{(\ell)}, t)) \mathbb{I}(t \in C_i) \bar{f}(\hat{k}^{(0)}, \bar{k}^{(\ell)}, \chi^{(\ell)}, \bar{n}^{(\ell)}) \frac{r^2}{\cos \theta} d\bar{k}^{(\ell)} d\hat{k}^{(0)} \quad (40)$$

where the inner sum covers all possible sequences of interaction types of depth ℓ , including diffuse reflections. The path $p(s, \hat{k}^{(0)}, \bar{k}^{(\ell)}, \chi^{(\ell)}, t)$ represents a path of depth ℓ from the source s to the target $t \in \mathcal{M}$. This path is defined by the initial propagation direction $\hat{k}^{(0)}$, the sequence of scattered ray directions $\bar{k}^{(\ell)}$, and the sequence of interaction types $\chi^{(\ell)}$. Compared to (39), the paths depend on the directions of propagation of the scattered waves, which are not solely determined by the initial propagation directions and interaction types when diffuse reflection occurs. In (40), $f(\cdot)$ in (37) is extended to include diffuse reflections as follows:

$$f(\chi, \hat{k}_i, \hat{k}_s, \hat{n}) = \begin{cases} \delta\left(\hat{k}_s - \left(\hat{k}_i - 2(\hat{n}^\top \hat{k}_i)\hat{n}\right)\right) & \text{if } \chi = \mathcal{R} \\ \delta\left(\hat{k}_s - \hat{k}_i\right) & \text{if } \chi = \mathcal{T} \\ \mathbb{I}\left(\hat{k}_i^\top \hat{n} > 0\right) & \text{if } \chi = \mathcal{S} \end{cases} \quad (41)$$

where $\mathbb{I}(\cdot)$ is the indicator function, which equals 1 if the condition is true and 0 otherwise. Note that in (40), r is the length of the ray tube impinging on the measurement plane, which, in the case of paths including diffuse reflections, corresponds to the distance from the intersection with the measurement plane to the last diffuse reflection point.

In practice, we estimate C_i using Monte Carlo integration. Similar to the path solver, we employ random sampling of interaction types (see Section 3.1.2) and limit the integration to paths with a maximum depth of $L < \infty$:

$$C_i = \frac{1}{|C|} \sum_{\ell=0}^L \Pr(\chi^{(\ell)}) \sum_{\chi^{(\ell)} \in \mathcal{I}^\ell} \int_{\Omega} \int_{\Omega^\ell} e(p(s, \hat{k}^{(0)}, \bar{k}^{(\ell)}, \chi^{(\ell)}, t)) \mathbb{I}(t \in C_i) \bar{f}(\hat{k}^{(0)}, \bar{k}^{(\ell)}, \chi^{(\ell)}, \bar{n}^{(\ell)}) \frac{r^2}{\Pr(\chi^{(\ell)}) \cos \theta} d\bar{k}^{(\ell)} d\hat{k}^{(0)} \quad (42)$$

Assuming N_S samples for Monte Carlo integration, we sample the unit sphere around the source s using N_S samples, resulting in initial ray tubes with a solid angle of approximately $4\pi/N_S$. For diffuse reflection points, we emit a single ray on the hemisphere, leading to ray tubes with a solid angle of 2π . This results in the following estimator

$$\widehat{C}_i = \frac{1}{|C|} \frac{4\pi}{N_S} \sum_{n=1}^{N_S} \sum_{\ell=0}^L e(p(s, \hat{k}_n^{(0)}, \bar{k}_n^{(\ell)}, \chi_n^{(\ell)}, t_n)) \mathbb{I}(t_n \in C_i) \frac{r_n^2 \left(\prod_{\ell'=1}^{\ell} \omega_n^{(\ell')} \right)}{\Pr(\chi_n) \cos \theta_n}. \quad (43)$$

Here, the subscript n denotes the n -th path, and $\omega_n^{(\ell')}$ represents the solid angles of the ray tubes corresponding to sampling the ℓ' -th integral:

$$\omega_n^{(\ell')} = \begin{cases} 2\pi & \text{if } \chi_n^{(\ell')} = \mathcal{S} \\ 1 & \text{otherwise.} \end{cases} \quad (44)$$

In the case of diffuse reflections, the multiplication by the ray tube solid angle is already applied through the computation of the scattered electric field (123). It has been included here for the sake of clarity. As we will see in the next section, this estimator can be efficiently computed using a single SBR loop.

4.1.1. Non-Coherent vs. Coherent Radio Maps

The proposed definition of radio maps involves a non-coherent summation of path gains (35). As a result, it does not capture fast fading effects, which result from the constructive and destructive interference of multiple paths. Figure 24 demonstrates this by comparing three scenarios: a radio map generated using the radio map solver (Section 4.2), a radio map created by non-coherently summing the path coefficients (4) calculated with the path solver (Section 3) by placing a single target at the center of each measurement cell, and a radio map produced by coherently summing the path coefficients (4) calculated with the path solver, also by placing a single target at the center of each measurement cell. The radio map obtained using the radio map solver aligns with the non-coherent summation of path coefficients calculated using the path solver, as anticipated. However, it does not account for fast fading effects, which requires the coherent summation of path coefficients.

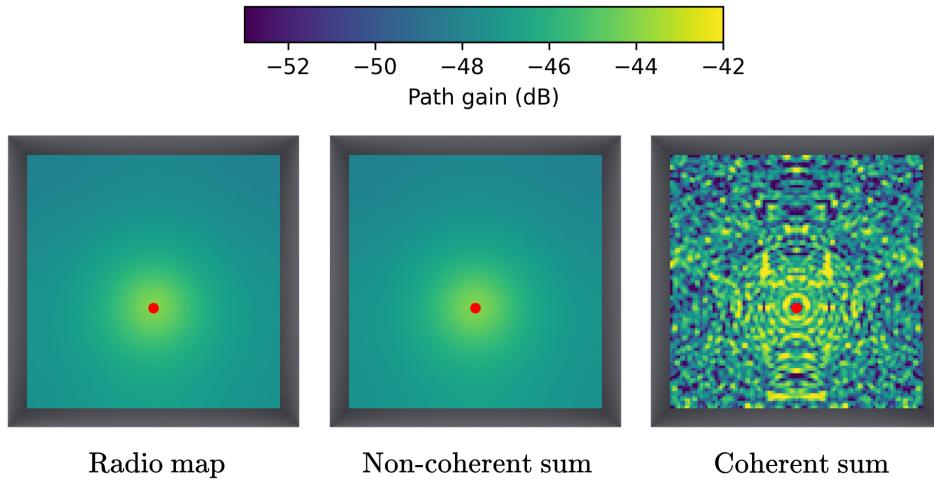


Figure 24: Comparison of radio maps: one computed using the radio map solver, one by non-coherently summing path coefficients from the path solver, and one by coherently summing path coefficients from the path solver. The transmitter is depicted as a red ball. All radio maps are computed with a maximum depth of $L = 3$, cell sizes of 10×10 cm, using isotropic antenna patterns on both the transmitter and receiver, and for the “box” scene built in Sionna RT. The “non-coherent sum” and “coherent sum” radio maps are generated by placing a single target point at the center of each measurement cell.

4.1.2. Handling Multiple Transmit Antennas

The Sionna RT radio map solver supports antenna arrays of any size at the transmitter and allows for user-defined precoding vectors. However, it only supports synthetic arrays. In this setup, paths are generated from a single source located at the center of the array, and the channel response for each individual antenna is calculated by applying the appropriate phase shifts synthetically. This approximation holds true as long as the array size is small compared to the distance from the transmitter to the measurement plane and scatterers.

Using synthetic arrays enables efficient scaling with respect to the number of antennas. To see this, let N_A^{tx} denote the number of antennas at the transmitter, and let $\mathbf{u}_P \in \mathbb{C}^{N_A^{\text{tx}}}$ be the precoding vector. For a given path connecting the source to the measurement plane, let $\mathbf{E} \in \mathbb{C}^2$ represent the electric field observed at the measurement plane. By employing a synthetic array, the corresponding electric fields for each antenna element are obtained by applying the array response vector (30), denoted as \mathbf{u}_A^{tx} , leading to

$$\mathbf{E}_A = \mathbf{E} (\mathbf{u}_A^{\text{tx}})^T. \quad (45)$$

Here, \mathbf{E}_A is a matrix of size $2 \times N_A^{\text{tx}}$, where each column represents the electric fields radiated by each transmit

antenna. The contribution of this path to the radio map is given by

$$\|\mathbf{E}_A \mathbf{u}_P\|_2^2. \quad (46)$$

This expression allows the path's contribution to the radio map to be represented as

$$\|\mathbf{E}_A \mathbf{u}_P\|_2^2 = \left\| \left(\mathbf{E} (\mathbf{u}_A^{\text{tx}})^T \mathbf{u}_P \right) \right\|_2^2 = \|\mathbf{E}\alpha\|_2^2 \quad (47)$$

where $\alpha = (\mathbf{u}_A^{\text{tx}})^T \mathbf{u}_P$ is a scalar. Note that α depends on the departure direction of the path from the transmitter and thus varies for paths originating from the source. However, it remains independent of the array size. By precomputing α for each path, the effect of the transmitter array can be efficiently simulated by multiplying the electric field of each path by α once.

4.2. SBR-based Radio Map Solver

The Sionna RT built-in radio map solver calculates the estimator (43) using a single **SBR** loop. Unlike the path solver described in Section 3, it does not require an image method-based algorithm, as it only needs to test intersections with the measurement plane rather than connecting paths to precisely located target points. Additionally, the electric field for each ray tube is computed dynamically during the **SBR** loop as the paths are traced.

Algorithm 3 outlines the SBR loop utilized by the Sionna RT radio map solver. While the algorithm is presented for a single path n , the radio map solver processes all paths in parallel. The process begins by sampling the departure direction for each ray using a Fibonacci lattice, as detailed in Section 3.1.1. The electric field is initialized using the transmit pattern evaluated at the path's departure direction $\hat{\mathbf{k}}_n^{(0)}$. The ray tube length is initially set to zero, and its solid angle is set to $4\pi/N_S$, since the N_S initial rays have departure directions on the unit sphere. The radio map is initialized to zero, and the path is marked as active. The **SBR** loop continues to trace the paths through the scene until a user-defined maximum depth $L \in \mathbb{N}$ is reached or the ray is deactivated. The loop's first step involves checking for intersections with the measurement plane. If an intersection is detected, the radio map is updated using the function `Add_To_Radio_Map`. The ray-plane intersection-checking function is not detailed in this document but is assumed to return the position of the intersection point \mathbf{v}_{mp} . The function `Add_To_Radio_Map` accumulates contributions to the radio map in the vector $\mathbf{C} = [C_1, \dots, C_{N_M}]$ at the intersected cell entry, as described in (43). Note that by iterating over path depth $\ell = 0, \dots, L$ and adding contributions to the radio map at every iteration, the radio map solver accounts for paths of all depths, from **LoS** to the maximum depth. The ray tube length is updated with the distance between the last intersection point and the measurement plane intersection point. Additionally, $\hat{\mathbf{n}}_{\text{mp}}$ represents the normal vector to the measurement plane, and $|\hat{\mathbf{k}}^T \hat{\mathbf{n}}_{\text{mp}}|$ is the cosine of the angle between the ray direction and the measurement plane normal. The weight α_n accounts for the transmitter array, as explained in Section 4.1.2. Next, the algorithm checks for intersections with the scene. If no intersection is found, the ray is deactivated as it has exited the scene. Otherwise, an interaction type $\chi_n^{(\ell)}$ is sampled using the probability distribution from Section 3.1.2, and the radio material is evaluated for the sampled interaction type. This results in the transfer matrix $\mathbf{T}_n^{(\ell)}$ and the scattering direction $\hat{\mathbf{k}}_n^{(\ell)}$. Unlike Algorithm 2, the function `Sample_Evaluate_Interaction` samples the interaction type, scattered direction, and evaluates the radio material. The electric field is updated by applying the transfer matrix $\mathbf{T}_n^{(\ell)}$ and dividing by the probability of the sampled interaction type. If the interaction type is diffuse, the ray tube length is reset to zero, and the solid angle weight is set to 2π , as diffuse reflection points act as new point sources, and as a single ray is emitted in the hemisphere containing the incident direction. Note that in the case of diffuse reflection, the matrix $\mathbf{T}_n^{(\ell)}$ applies the scaling by $\omega_n^{(\ell-1)}$. The **SBR** loop concludes by updating the ray tube length with the distance between the last and current intersection points. Finally, the radio map computation is completed by scaling by $\lambda^2/(4\pi)^2$ according to (35), and by the surface area of measurement cells, following the radio map definition in (33).

Figure 25b illustrates the computation time needed to generate a radio map for a single source as the number of samples N_S and the number of transmit antennas N_A^{tx} are varied. The setup, depicted in Figure 25a, employs the “simple street canyon” scene built into Sionna RT. The red ball indicates the source, positioned 20 meters above the ground. The radio map in Figure 25a is calculated using 10^8 samples and a linear array of 512

Algorithm 3 Radio map solver SBR loop

```

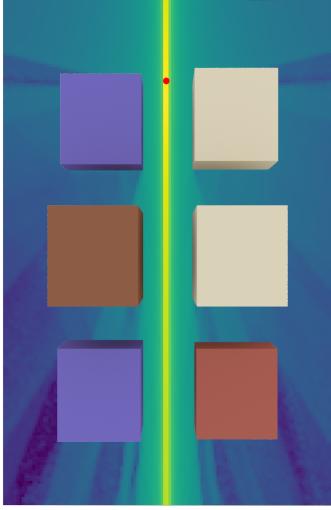
1:  $\hat{\mathbf{k}}_n^{(0)} \leftarrow \text{FIBONACCI\_LATTICE}(N_S)$                                 ▷ Sample departure direction
2:  $\mathbf{E}_n^{(0)} \leftarrow \text{TX\_PATTERN}(\hat{\mathbf{k}}_n^{(0)})$                             ▷ Electric field
3:  $r_n^{(0)} \leftarrow 0$                                                                ▷ Ray tube length
4:  $\omega_n^{(0)} \leftarrow \frac{4\pi}{N_S}$                                             ▷ Ray tube solid angle
5:  $\mathbf{C} = (\hat{C}_1, \dots, \hat{C}_{N_M}) = (0, \dots, 0)$                          ▷ Radio map
6:  $\text{active}_n \leftarrow \text{True}$                                                  ▷ Path active flag
7:  $\mathbf{v}_n^{(0)} \leftarrow \mathbf{s}$                                                  ▷ Path vertex
8:  $\ell \leftarrow 1$                                                                ▷ Path depth
9: while  $\ell \leq L$  and  $\text{active}_n$  do
10:    $\text{mp\_inter}_n, \mathbf{v}_{\text{mp},n}^{(\ell)} \leftarrow \text{MP\_INTERSECTION}(\mathbf{v}_n^{(\ell-1)}, \hat{\mathbf{k}}_n^{(\ell-1)})$ 
11:   if  $\text{mp\_inter}_n = \text{True}$  then
12:      $\mathbf{C} \leftarrow \text{ADD\_TO\_RADIO\_MAP}(\mathbf{C}, \mathbf{E}_n^{(\ell-1)}, \hat{\mathbf{k}}_n^{(\ell-1)}, r_n^{(\ell-1)}, \omega_n^{(\ell-1)}, \mathbf{v}_{\text{mp},n}^{(\ell)}, \mathbf{v}_n^{(\ell-1)})$ 
13:   end if
14:    $\text{sc\_inter}_n, o_n^{(\ell)}, \mathbf{v}_n^{(\ell)} \leftarrow \text{SCENE\_INTERSECTION}(\mathbf{v}_n^{(\ell-1)}, \hat{\mathbf{k}}_n^{(\ell-1)})$ 
15:    $\text{active}_n \leftarrow \text{active}_n$  and  $\text{sc\_inter}_n$ 
16:   if  $\text{active}_n = \text{True}$  then
17:      $\mathbf{T}_n^{(\ell)}, \hat{\mathbf{k}}_n^{(\ell)}, \chi_n^{(\ell)} \leftarrow \text{SAMPLE\_EVALUATE\_INTERACTION}(o_n^{(\ell)}, \mathbf{E}_n^{(\ell-1)}, \hat{\mathbf{k}}_n^{(\ell-1)}, \omega_n^{(\ell-1)})$ 
18:      $\mathbf{E}_n^{(\ell)} \leftarrow \frac{\mathbf{T}_n^{(\ell)} \mathbf{E}_n^{(\ell-1)}}{\sqrt{q_n^{(\ell)}(\chi_n^{(\ell)})}}$ 
19:     if  $\chi_n^{(\ell)} = \mathcal{S}$  then
20:        $\omega_n^{(\ell)} \leftarrow 2\pi$ 
21:        $r_n^{(\ell-1)} \leftarrow 0$ 
22:     end if
23:      $r_n^{(\ell)} \leftarrow r_n^{(\ell-1)} + \|\mathbf{v}_n^{(\ell)} - \mathbf{v}_n^{(\ell-1)}\|_2$ 
24:      $\ell \leftarrow \ell + 1$ 
25:   end if
26: end while
27:  $\mathbf{C} \leftarrow \frac{\lambda^2}{(4\pi)^2 |\mathbf{C}|} \mathbf{C}$ 
28: return  $\mathbf{C}$ 
29:
30: function  $\text{ADD\_TO\_RADIO\_MAP}(\mathbf{C}, \mathbf{E}, \hat{\mathbf{k}}, r, \omega, \mathbf{v}_{\text{mp}}, \mathbf{v})$ 
31:    $d \leftarrow r + \|\mathbf{v}_{\text{mp}} - \mathbf{v}\|_2$ 
32:    $i \leftarrow \text{GET\_MAP\_INDEX}(\mathbf{v}_{\text{mp}})$ 
33:    $\mathbf{C}_i \leftarrow \mathbf{C}_i + \frac{\omega d^2}{|\hat{\mathbf{k}}^\top \hat{\mathbf{n}}_{\text{mp}}|} \|\alpha_n \mathbf{E}\|_2^2$ 
34:   return  $\mathbf{C}$ 
35: end function

```

transmit antennas. Notably, the radio map solver can compute radio maps with 10^9 samples on a single NVIDIA RTX 4090 GPU. The computation time increases with both the number of samples and antennas, exceeding one second only for the most demanding computations. It is important to note that while the computation time increases with the number of antennas due to the precomputation of the array weighting factor α , it does not increase the memory footprint of the radio map solver (see Section 4.1.2).

Handling of Dual-Polarized Transmit Antennas

The radio map solver supports dual-polarized transmit antennas by considering two independent electric field vectors for each path. These are processed as if they were separate antenna elements (see Section 4.1.2).



(a) Scene with the radio map.

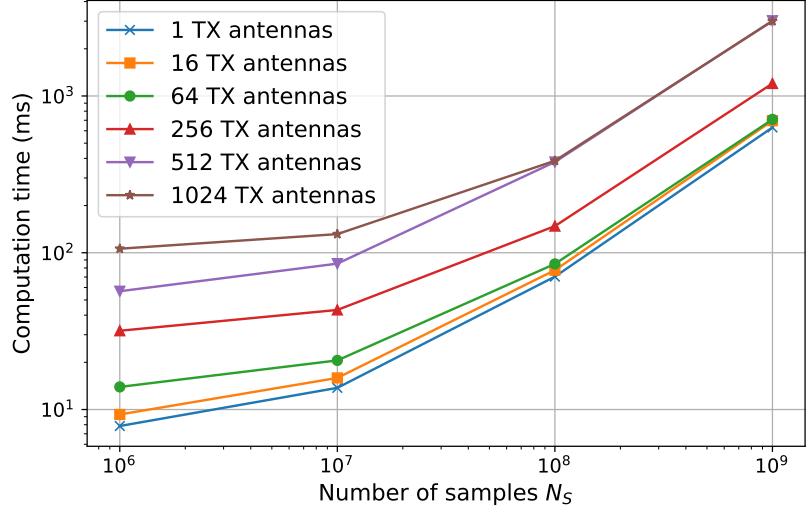
(b) Computation time vs. number of samples N_S .

Figure 25: Computation time for calculating radio maps using an NVIDIA RTX 4090 GPU for a single transmitter with a linear array of 1 to 1024 antennas.

Scaling with the Number of Measurement Cells

A key benefit of the radio map solver is that the compute required to generate the radio map remains constant regardless of the number of measurement cells. However, increasing the number of measurement cells will naturally require more memory for storage. Additionally, when the size of the measurement cells is reduced, it may be necessary to increase the number of samples N_S to maintain the accuracy of the estimator (43), as fewer paths will intersect with each measurement cell, resulting in noisier radio maps.

Early Termination with Russian Roulette and Gain Thresholding

The radio map solver allows for early deactivation of paths using two techniques: Russian roulette and gain thresholding. With gain thresholding, a path with ray tube length r is deactivated when the squared amplitude of the electric field $\|\mathbf{E}\|_2^2$ falls below a user-defined threshold. With Russian roulette, at each iteration, a path continues with a probability given by

$$p_{rr} = \min \left(r^2 \|\mathbf{E}\|_2^2, p_{rr,\max} \right) \quad (48)$$

where $p_{rr,\max}$ is the maximum probability, set by the user, for keeping the path active. Both features can be enabled simultaneously and are applied only after a user-defined depth is reached.

Extension to other Types of Radio Maps

The Sionna RT solver currently supports the calculation of channel gain maps, which are radio maps that provide an estimate of the channel gain for each cell. It is possible to extend the solver to calculate other types of radio maps, such as root-mean-square delay spread maps and angular spread maps [18].

A. Primer on Electromagnetism

This section provides useful background for the general understanding of ray tracing for wireless propagation modeling. In particular, our goal is to provide a concise definition of a “channel impulse response” between a transmitting and receiving antenna, as in [19, Ch. 2 & 3].

A.1. Coordinate System, Rotations, and Vector Fields

We consider a **global coordinate system (GCS)** with Cartesian canonical basis $\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}$. The spherical unit vectors are defined as

$$\begin{aligned}\hat{\mathbf{r}}(\theta, \varphi) &= \sin(\theta) \cos(\varphi) \hat{\mathbf{x}} + \sin(\theta) \sin(\varphi) \hat{\mathbf{y}} + \cos(\theta) \hat{\mathbf{z}} \\ \hat{\boldsymbol{\theta}}(\theta, \varphi) &= \cos(\theta) \cos(\varphi) \hat{\mathbf{x}} + \cos(\theta) \sin(\varphi) \hat{\mathbf{y}} - \sin(\theta) \hat{\mathbf{z}} \\ \hat{\boldsymbol{\varphi}}(\theta, \varphi) &= -\sin(\varphi) \hat{\mathbf{x}} + \cos(\varphi) \hat{\mathbf{y}}.\end{aligned}\quad (49)$$

For an arbitrary unit norm vector $\hat{\mathbf{v}} = (x, y, z)$, the zenith and azimuth angles θ and φ can be computed as

$$\begin{aligned}\theta &= \cos^{-1}(z) \\ \varphi &= \text{atan2}(y, x)\end{aligned}\quad (50)$$

where $\text{atan2}(y, x)$ is the two-argument inverse tangent function [20]. As any vector uniquely determines θ and φ , we sometimes also write $\hat{\boldsymbol{\theta}}(\hat{\mathbf{v}})$ and $\hat{\boldsymbol{\varphi}}(\hat{\mathbf{v}})$ instead of $\hat{\boldsymbol{\theta}}(\theta, \varphi)$ and $\hat{\boldsymbol{\varphi}}(\theta, \varphi)$.

A 3D rotation with yaw, pitch, and roll angles α , β , and γ , respectively, is expressed by the matrix

$$\mathbf{R}(\alpha, \beta, \gamma) = \mathbf{R}_z(\alpha) \mathbf{R}_y(\beta) \mathbf{R}_x(\gamma) \quad (51)$$

where $\mathbf{R}_z(\alpha)$, $\mathbf{R}_y(\beta)$, and $\mathbf{R}_x(\gamma)$ are rotation matrices around the z , y , and x axes, respectively, which are defined as

$$\begin{aligned}\mathbf{R}_z(\alpha) &= \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ \mathbf{R}_y(\beta) &= \begin{pmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{pmatrix} \\ \mathbf{R}_x(\gamma) &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\gamma) & -\sin(\gamma) \\ 0 & \sin(\gamma) & \cos(\gamma) \end{pmatrix}.\end{aligned}\quad (52)$$

A closed-form expression for $\mathbf{R}(\alpha, \beta, \gamma)$ can be found in [17, Sec. 7.1-4]. The inverse rotation is simply defined by $\mathbf{R}^{-1}(\alpha, \beta, \gamma) = \mathbf{R}^T(\alpha, \beta, \gamma)$. A vector \mathbf{x} defined in a first coordinate system is represented in a second coordinate system rotated by $\mathbf{R}(\alpha, \beta, \gamma)$ with respect to the first one as $\mathbf{x}' = \mathbf{R}^T(\alpha, \beta, \gamma) \mathbf{x}$. If a point in the first coordinate system has spherical angles (θ, φ) , the corresponding angles (θ', φ') in the second coordinate system can be found to be

$$\begin{aligned}\theta' &= \cos^{-1}(\hat{\mathbf{z}}^T \mathbf{R}^T(\alpha, \beta, \gamma) \hat{\mathbf{r}}(\theta, \varphi)) \\ \varphi' &= \arg((\hat{\mathbf{x}} + j\hat{\mathbf{y}})^T \mathbf{R}^T(\alpha, \beta, \gamma) \hat{\mathbf{r}}(\theta, \varphi)).\end{aligned}\quad (53)$$

For a vector field $\mathbf{F}'(\theta', \varphi')$ expressed in local spherical coordinates

$$\mathbf{F}'(\theta', \varphi') = F_{\theta'}(\theta', \varphi') \hat{\boldsymbol{\theta}}'(\theta', \varphi') + F_{\varphi'}(\theta', \varphi') \hat{\boldsymbol{\varphi}}'(\theta', \varphi') \quad (54)$$

that are rotated by $\mathbf{R} = \mathbf{R}(\alpha, \beta, \gamma)$ with respect to the **GCS**, the spherical field components in the **GCS** can be expressed as

$$\begin{bmatrix} F_\theta(\theta, \varphi) \\ F_\varphi(\theta, \varphi) \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{\theta}}(\theta, \varphi)^T \mathbf{R} \hat{\boldsymbol{\theta}}'(\theta', \varphi') & \hat{\boldsymbol{\theta}}(\theta, \varphi)^T \mathbf{R} \hat{\boldsymbol{\varphi}}'(\theta', \varphi') \\ \hat{\boldsymbol{\varphi}}(\theta, \varphi)^T \mathbf{R} \hat{\boldsymbol{\theta}}'(\theta', \varphi') & \hat{\boldsymbol{\varphi}}(\theta, \varphi)^T \mathbf{R} \hat{\boldsymbol{\varphi}}'(\theta', \varphi') \end{bmatrix} \begin{bmatrix} F_{\theta'}(\theta', \varphi') \\ F_{\varphi'}(\theta', \varphi') \end{bmatrix} \quad (55)$$

so that

$$\mathbf{F}(\theta, \varphi) = F_\theta(\theta, \varphi) \hat{\boldsymbol{\theta}}(\theta, \varphi) + F_\varphi(\theta, \varphi) \hat{\boldsymbol{\varphi}}(\theta, \varphi). \quad (56)$$

Sometimes, it is also useful to find the rotation matrix that maps a unit vector $\hat{\mathbf{a}}$ to $\hat{\mathbf{b}}$. This can be achieved with the help of Rodrigues' rotation formula [21] which defines the matrix

$$\mathbf{R}(\hat{\mathbf{a}}, \hat{\mathbf{b}}) = \mathbf{I} + \sin(\theta)\mathbf{K} + (1 - \cos(\theta))\mathbf{K}^2 \quad (57)$$

where

$$\mathbf{K} = \begin{bmatrix} 0 & -\hat{k}_z & \hat{k}_y \\ \hat{k}_z & 0 & -\hat{k}_x \\ -\hat{k}_y & \hat{k}_x & 0 \end{bmatrix} \quad (58)$$

$$\hat{\mathbf{k}} = \frac{\hat{\mathbf{a}} \times \hat{\mathbf{b}}}{\|\hat{\mathbf{a}} \times \hat{\mathbf{b}}\|_2} \quad (59)$$

$$\theta = \hat{\mathbf{a}}^\top \hat{\mathbf{b}} \quad (60)$$

such that $\mathbf{R}(\hat{\mathbf{a}}, \hat{\mathbf{b}})\hat{\mathbf{a}} = \hat{\mathbf{b}}$.

A.2. Planar Time-Harmonic Waves

A time-harmonic planar electric wave $\mathbf{E}(\mathbf{x}, t) \in \mathbb{C}^3$ traveling in a homogeneous medium with wave vector $\mathbf{k} \in \mathbb{C}^3$ can be described at position $\mathbf{x} \in \mathbb{R}^3$ and time t as

$$\mathbf{E}(\mathbf{x}, t) = \mathbf{E}_0 e^{j(\omega t - \mathbf{k}^\text{H} \mathbf{x})} \quad (61)$$

$$= \mathbf{E}(\mathbf{x}) e^{j\omega t} \quad (62)$$

where $\mathbf{E}_0 \in \mathbb{C}^3$ is the field phasor. The wave vector can be decomposed as $\mathbf{k} = k\hat{\mathbf{k}}$, where $\hat{\mathbf{k}}$ is a unit norm vector, $k = \omega\sqrt{\varepsilon\mu}$ is the wave number, and $\omega = 2\pi f$ is the angular frequency. The permittivity ε and permeability μ are defined as

$$\varepsilon = \eta\varepsilon_0 \quad (63)$$

$$\mu = \mu_r\mu_0 \quad (64)$$

where η and ε_0 are the complex relative and vacuum permittivities, μ_r and μ_0 are the relative and vacuum permeabilities, and σ is the conductivity. The complex relative permittivity η is given as

$$\eta = \varepsilon_r - j\frac{\sigma}{\varepsilon_0\omega} \quad (65)$$

where ε_r is the real relative permittivity of a non-conducting dielectric.

With these definitions, the speed of light is given as [22, Eq. 4-28d]

$$c = \frac{1}{\sqrt{\varepsilon_0\varepsilon_r\mu}} \left\{ \frac{1}{2} \left(\sqrt{1 + \left(\frac{\sigma}{\omega\varepsilon_0\varepsilon_r} \right)^2} + 1 \right) \right\}^{-\frac{1}{2}} \quad (66)$$

where the factor in curly brackets vanishes for non-conducting materials. The speed of light in vacuum is denoted $c_0 = \frac{1}{\sqrt{\varepsilon_0\mu_0}}$ and the vacuum wave number $k_0 = \frac{\omega}{c_0}$. In conducting materials, the wave number is complex which translates to propagation losses.

The associated magnetic field $\mathbf{H}(\mathbf{x}, t) \in \mathbb{C}^3$ is

$$\mathbf{H}(\mathbf{x}, t) = \frac{\hat{\mathbf{k}} \times \mathbf{E}(\mathbf{x}, t)}{Z} = \mathbf{H}(\mathbf{x}) e^{j\omega t} \quad (67)$$

where $Z = \sqrt{\mu/\varepsilon}$ is the wave impedance. The vacuum impedance is denoted by $Z_0 = \sqrt{\mu_0/\varepsilon_0} \approx 376.73 \Omega$.

The time-averaged Poynting vector is defined as

$$\mathbf{S}(\mathbf{x}) = \frac{1}{2} \Re \{ \mathbf{E}(\mathbf{x}) \times \mathbf{H}(\mathbf{x}) \} = \frac{1}{2} \Re \left\{ \frac{1}{Z} \right\} \|\mathbf{E}(\mathbf{x})\|_2^2 \hat{\mathbf{k}} \quad (68)$$

which describes the directional energy flux [W m^{-2}], i.e. energy transfer per unit area per unit time. Note that the actual electromagnetic waves are the real parts of $\mathbf{E}(\mathbf{x}, t)$ and $\mathbf{H}(\mathbf{x}, t)$.

A.3. Far Field of a Transmitting Antenna

We assume that the electric far field of an antenna in free space can be described by a spherical wave originating from the center of the antenna:

$$\mathbf{E}(r, \theta, \varphi, t) = \mathbf{E}(r, \theta, \varphi) e^{j\omega t} = \mathbf{E}_0(\theta, \varphi) \frac{e^{-jk_0 r}}{r} e^{j\omega t} \quad (69)$$

where $\mathbf{E}_0(\theta, \varphi)$ is the electric field phasor, r is the distance (or radius), θ the zenith angle, and φ the azimuth angle. In contrast to a planar wave, the field strength decays as $1/r$.

The complex antenna field pattern $\mathbf{F}(\theta, \varphi)$ is defined as

$$\mathbf{F}(\theta, \varphi) = \frac{\mathbf{E}_0(\theta, \varphi)}{\max_{\theta, \varphi} \|\mathbf{E}_0(\theta, \varphi)\|_2}. \quad (70)$$

The time-averaged Poynting vector for such a spherical wave is

$$\mathbf{S}(r, \theta, \varphi) = \frac{1}{2Z_0} \|\mathbf{E}(r, \theta, \varphi)\|_2^2 \hat{\mathbf{r}} \quad (71)$$

where $\hat{\mathbf{r}}$ is the radial unit vector. It simplifies for an ideal isotropic antenna with input power P_T to

$$\mathbf{S}_{\text{iso}}(r, \theta, \varphi) = \frac{P_T}{4\pi r^2} \hat{\mathbf{r}}. \quad (72)$$

The antenna gain G is the ratio of the maximum radiation power density of the antenna in radial direction and that of an ideal isotropic radiating antenna:

$$G = \frac{\max_{\theta, \varphi} \|\mathbf{S}(r, \theta, \varphi)\|_2}{\|\mathbf{S}_{\text{iso}}(r, \theta, \varphi)\|_2} = \frac{2\pi}{Z_0 P_T} \max_{\theta, \varphi} \|\mathbf{E}_0(\theta, \varphi)\|_2^2. \quad (73)$$

One can similarly define a gain with directional dependency by ignoring the computation of the maximum the last equation:

$$G(\theta, \varphi) = \frac{2\pi}{Z_0 P_T} \|\mathbf{E}_0(\theta, \varphi)\|_2^2 = G \|\mathbf{F}(\theta, \varphi)\|_2^2. \quad (74)$$

If one uses in the last equation the radiated power $P = \eta_{\text{rad}} P_T$, where η_{rad} is the radiation efficiency, instead of the input power P_T , one obtains the directivity $D(\theta, \varphi)$. Both are related through $G(\theta, \varphi) = \eta_{\text{rad}} D(\theta, \varphi)$.

Antenna Pattern

Since $\mathbf{F}(\theta, \varphi)$ contains no information about the maximum gain G and $G(\theta, \varphi)$ does not carry any phase information, we define the “antenna pattern” $\mathbf{C}(\theta, \varphi)$ as

$$\mathbf{C}(\theta, \varphi) = \sqrt{G} \mathbf{F}(\theta, \varphi) \quad (75)$$

such that $G(\theta, \varphi) = \|\mathbf{C}(\theta, \varphi)\|_2^2$. Using the spherical unit vectors $\hat{\theta} \in \mathbb{R}^3$ and $\hat{\varphi} \in \mathbb{R}^3$, we can rewrite $\mathbf{C}(\theta, \varphi)$ as

$$\mathbf{C}(\theta, \varphi) = C_\theta(\theta, \varphi) \hat{\theta} + C_\varphi(\theta, \varphi) \hat{\varphi} \quad (76)$$

where $C_\theta(\theta, \varphi) \in \mathbb{C}$ and $C_\varphi(\theta, \varphi) \in \mathbb{C}$ are the “zenith pattern” and “azimuth pattern”, respectively.

Combining F and G , we can obtain the following expression of the electric far field

$$\mathbf{E}_T(r, \theta_T, \varphi_T) = \sqrt{\frac{P_T G_T Z_0}{2\pi}} \frac{e^{-jk_0 r}}{r} \mathbf{F}_T(\theta_T, \varphi_T) \quad (77)$$

where we have added the subscript T to all quantities that are specific to the transmitting antenna.

The input power P_T of an antenna with (conjugate matched) impedance Z_T , fed by a voltage source with complex amplitude V_T , is given by (see e.g., [23]).

$$P_T = \frac{|V_T|^2}{8\Re\{Z_T\}}. \quad (78)$$

Normalization of Antenna Patterns

The radiated power $\eta_{\text{rad}} P_T$ of an antenna can be obtained by integrating the Poynting vector over the surface of a closed sphere of radius r around the antenna:

$$\eta_{\text{rad}} P_T = \int_0^{2\pi} \int_0^\pi \mathbf{S}(r, \theta, \varphi)^T \hat{\mathbf{r}} r^2 \sin(\theta) d\theta d\varphi \quad (79)$$

$$= \int_0^{2\pi} \int_0^\pi \frac{1}{2Z_0} \|\mathbf{E}(r, \theta, \varphi)\|_2^2 r^2 \sin(\theta) d\theta d\varphi \quad (80)$$

$$= \frac{P_T}{4\pi} \int_0^{2\pi} \int_0^\pi G(\theta, \varphi) \sin(\theta) d\theta d\varphi. \quad (81)$$

We can see from the last equation that the directional gain of any antenna must satisfy

$$\int_0^{2\pi} \int_0^\pi G(\theta, \varphi) \sin(\theta) d\theta d\varphi = 4\pi \eta_{\text{rad}}. \quad (82)$$

A.4. Modeling of a Receiving Antenna

Although the transmitting antenna radiates a spherical wave $\mathbf{E}_T(r, \theta_T, \varphi_T)$, we assume that the receiving antenna observes a planar incoming wave \mathbf{E}_R that arrives from the angles θ_R and φ_R which are defined in the local spherical coordinates of the receiving antenna. The Poynting vector of the incoming wave \mathbf{S}_R is hence (71)

$$\mathbf{S}_R = -\frac{1}{2Z_0} \|\mathbf{E}_R\|_2^2 \hat{\mathbf{r}}(\theta_R, \varphi_R) \quad (83)$$

where $\hat{\mathbf{r}}(\theta_R, \varphi_R)$ is the radial unit vector in the spherical coordinate system of the receiver.

The aperture or effective area A_R of an antenna with gain G_R is defined as the ratio of the available received power P_R at the output of the antenna and the absolute value of the Poynting vector, i.e. the power density:

$$A_R = \frac{P_R}{\|\mathbf{S}_R\|_2} = G_R \frac{\lambda^2}{4\pi} \quad (84)$$

where $\frac{\lambda^2}{4\pi}$ is the aperture of an isotropic antenna. In the definition above, it is assumed that the antenna is ideally directed towards and polarization matched to the incoming wave. For an arbitrary orientation of the antenna (but still assuming polarization matching), we can define a direction dependent effective area

$$A_R(\theta_R, \varphi_R) = G_R(\theta_R, \varphi_R) \frac{\lambda^2}{4\pi}. \quad (85)$$

The available received power at the output of the antenna can be expressed as

$$P_R = \frac{|V_R|^2}{8\Re\{Z_R\}} \quad (86)$$

where Z_R is the impedance of the receiving antenna and V_R the open circuit voltage. We can now combine (86), (85), and (84) to obtain the following expression for the absolute value of the voltage $|V_R|$ assuming

matched polarization:

$$|V_R| = \sqrt{P_R 8\Re\{Z_R\}} \quad (87)$$

$$= \sqrt{\frac{\lambda^2}{4\pi} G_R(\theta_R, \varphi_R) \frac{8\Re\{Z_R\}}{2Z_0} \|\mathbf{E}_R\|_2^2} \quad (88)$$

$$= \sqrt{\frac{\lambda^2}{4\pi} G_R \frac{4\Re\{Z_R\}}{Z_0} \|\mathbf{F}_R(\theta_R, \varphi_R)\|_2 \|\mathbf{E}_R\|_2}. \quad (89)$$

By extension of the previous equation, we can obtain an expression for V_R which is valid for arbitrary polarizations of the incoming wave and the receiving antenna:

$$V_R = \sqrt{\frac{\lambda^2}{4\pi} G_R \frac{4\Re\{Z_R\}}{Z_0}} \mathbf{F}_R(\theta_R, \varphi_R)^H \mathbf{E}_R. \quad (90)$$

Recovering Friis Equation

In the case of free space propagation, we have $\mathbf{E}_R = \mathbf{E}_T(r, \theta_T, \varphi_T)$. Combining (90), (86), and (77), we obtain the following expression for the received power:

$$P_R = \left(\frac{\lambda}{4\pi r} \right)^2 G_R G_T P_T |\mathbf{F}_R(\theta_R, \varphi_R)^H \mathbf{F}_T(\theta_T, \varphi_T)|^2. \quad (91)$$

It is important that \mathbf{F}_R and \mathbf{F}_T are expressed in the same coordinate system for the last equation to make sense. For perfect orientation and polarization matching, we can recover the well-known Friis transmission equation

$$\frac{P_R}{P_T} = \left(\frac{\lambda}{4\pi r} \right)^2 G_R G_T. \quad (92)$$

A.5. General Propagation Path

A single propagation path consists of a sequence of scattering processes, where a scattering process can be anything that prevents the wave from propagating as in free space. This includes specular reflection, refraction, diffraction, and diffuse reflection. For each scattering process, one needs to compute a relationship between the incoming field at the scatter center and the created far field at the next scatter center or the receiving antenna. We can represent this cascade of scattering processes by a single matrix $\tilde{\mathbf{T}}$ that describes the transformation that the radiated field $\mathbf{E}_T(r, \theta_T, \varphi_T)$ undergoes until it reaches the receiving antenna:

$$\mathbf{E}_R = \sqrt{\frac{P_T G_T Z_0}{2\pi}} \tilde{\mathbf{T}} \mathbf{F}_T(\theta_T, \varphi_T). \quad (93)$$

Note that we have obtained this expression by replacing the free space propagation term $\frac{e^{-jk_0r}}{r}$ in (77) by the matrix $\tilde{\mathbf{T}}$. This requires that all quantities are expressed in the same coordinate system which is also assumed in the following expressions. Further, it is assumed that the matrix $\tilde{\mathbf{T}}$ includes the necessary coordinate transformations. In some cases, e.g., for diffuse reflection (see (123) in Section A.8), the matrix $\tilde{\mathbf{T}}$ depends on the incoming field and is not a linear transformation.

Plugging (93) into (90), we can obtain a general expression for the received voltage of a propagation path:

$$V_R = \sqrt{\left(\frac{\lambda}{4\pi} \right)^2 G_R G_T P_T 8\Re\{Z_R\} \mathbf{F}_R(\theta_R, \varphi_R)^H \tilde{\mathbf{T}} \mathbf{F}_T(\theta_T, \varphi_T)}. \quad (94)$$

If the electromagnetic wave arrives at the receiving antenna over N propagation paths, we can simply add the received voltages from all paths to obtain

$$V_R = \sqrt{\left(\frac{\lambda}{4\pi}\right)^2 G_R G_T P_T 8\Re\{Z_R\}} \sum_{n=1}^N \mathbf{F}_R(\theta_{R,i}, \varphi_{R,i})^H \tilde{\mathbf{T}}_i \mathbf{F}_T(\theta_{T,i}, \varphi_{T,i}) \quad (95)$$

$$= \sqrt{\left(\frac{\lambda}{4\pi}\right)^2 P_T 8\Re\{Z_R\}} \sum_{n=1}^N \mathbf{C}_R(\theta_{R,i}, \varphi_{R,i})^H \tilde{\mathbf{T}}_i \mathbf{C}_T(\theta_{T,i}, \varphi_{T,i}) \quad (96)$$

where all path-dependent quantities carry the subscript i . Note that the matrices $\tilde{\mathbf{T}}_i$ also ensure appropriate scaling so that the total received power can never be larger than the transmit power.

A.6. Frequency and Impulse Response

The channel frequency response $H(f)$ at frequency $f = \frac{c}{\lambda}$ is defined as the ratio between the received voltage and the voltage at the input to the transmitting antenna:

$$H(f) = \frac{V_R}{V_T} = \frac{V_R}{|V_T|} \quad (97)$$

where it is assumed that the input voltage has zero phase.

It is useful to separate phase shifts due to wave propagation from the transfer matrices $\tilde{\mathbf{T}}_i$. If we denote by r_i the total length of path i with average propagation speed c_i , the path delay is $\tau_i = r_i/c_i$. We can now define the new transfer matrix

$$\mathbf{T}_i = \tilde{\mathbf{T}}_i e^{j2\pi f \tau_i}. \quad (98)$$

Using (78) and (98) in (95) while assuming equal real parts of both antenna impedances, i.e. $\Re\{Z_T\} = \Re\{Z_R\}$ (which is typically the case), we obtain the final expression for the channel frequency response:

$$H(f) = \underbrace{\sum_{i=1}^N \frac{\lambda}{4\pi} \mathbf{C}_R(\theta_{R,i}, \varphi_{R,i})^H \mathbf{T}_i \mathbf{C}_T(\theta_{T,i}, \varphi_{T,i})}_{\triangleq a_i} e^{-j2\pi f \tau_i} \quad (99)$$

Taking the inverse Fourier transform, we finally obtain the channel impulse response

$$h(\tau) = \int_{-\infty}^{\infty} H(f) e^{j2\pi f \tau} df = \sum_{i=1}^N a_i \delta(\tau - \tau_i) \quad (100)$$

The baseband equivalent channel impulse response is then defined as [24, Eq. 2.28]:

$$h_b(\tau) = \sum_{i=1}^N \underbrace{a_i e^{-j2\pi f \tau_i}}_{\triangleq a_i^b} \delta(\tau - \tau_i). \quad (101)$$

A.7. Specular Reflection and Refraction

When a plane wave hits a plane interface which separates two materials, e.g., air and concrete, a part of the wave gets reflected and the other refracted, i.e. it propagates into the other material. We assume in the following description that both materials are uniform non-magnetic dielectrics, i.e. $\mu_r = 1$, and follow the definitions as in [25]. The incoming wave phasor \mathbf{E}_i is expressed by two arbitrary orthogonal polarization components, i.e.

$$\mathbf{E}_i = E_{i,s} \hat{\mathbf{e}}_{i,s} + E_{i,p} \hat{\mathbf{e}}_{i,p} \quad (102)$$

which are both orthogonal to the incident wave vector, i.e. $\hat{\mathbf{e}}_{i,s}^T \hat{\mathbf{e}}_{i,p} = \hat{\mathbf{e}}_{i,s}^T \hat{\mathbf{k}}_i = \hat{\mathbf{e}}_{i,p}^T \hat{\mathbf{k}}_i = 0$.

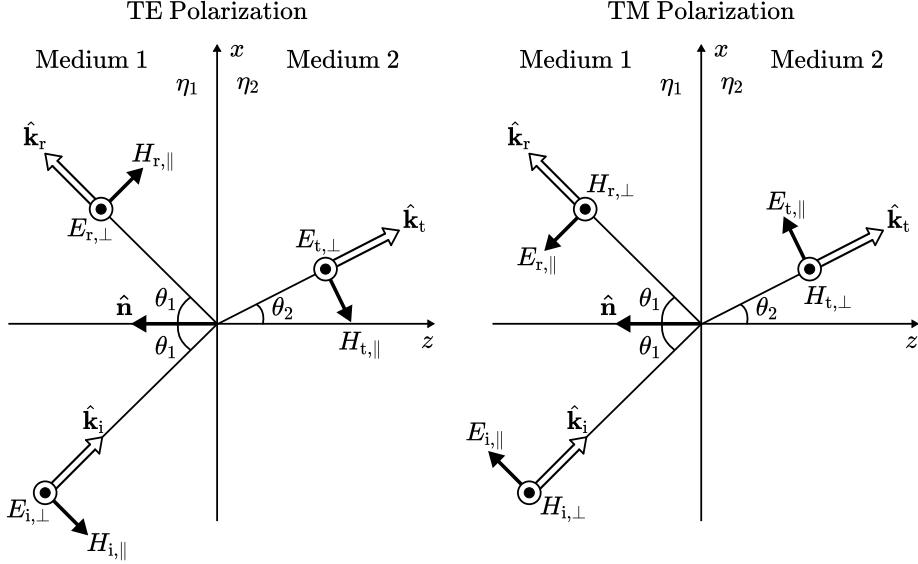


Figure 26: Reflection and refraction of a plane wave at a plane interface between two materials.

Figure 26 shows reflection and refraction of the incoming wave at the plane interface between two materials with relative permittivities η_1 and η_2 . The coordinate system is chosen such that the wave vectors of the incoming, reflected, and transmitted waves lie within the plane of incidence, which is chosen to be the x-z plane. The normal vector of the interface \hat{n} is pointing toward the negative z axis. The incoming wave is must be represented in a different basis, i.e. in the form two different orthogonal polarization components $E_{i,\perp}$ and $E_{i,\parallel}$, i.e.

$$\mathbf{E}_i = E_{i,\perp} \hat{\mathbf{e}}_{i,\perp} + E_{i,\parallel} \hat{\mathbf{e}}_{i,\parallel} \quad (103)$$

where the former is orthogonal to the plane of incidence and called transverse electric (TE) polarization (left), and the latter is parallel to the plane of incidence and called transverse magnetic (TM) polarization (right). In the following, we adopt the convention that all transverse components are coming out of the figure (indicated by the \odot symbol). One can easily verify that the following relationships must hold:

$$\begin{aligned} \hat{\mathbf{e}}_{i,\perp} &= \frac{\hat{\mathbf{k}}_i \times \hat{\mathbf{n}}}{\|\hat{\mathbf{k}}_i \times \hat{\mathbf{n}}\|_2} \\ \hat{\mathbf{e}}_{i,\parallel} &= \hat{\mathbf{e}}_{i,\perp} \times \hat{\mathbf{k}}_i \\ \begin{bmatrix} E_{i,\perp} \\ E_{i,\parallel} \end{bmatrix} &= \begin{bmatrix} \hat{\mathbf{e}}_{i,\perp}^\top \hat{\mathbf{e}}_{i,s} & \hat{\mathbf{e}}_{i,\perp}^\top \hat{\mathbf{e}}_{i,p} \\ \hat{\mathbf{e}}_{i,\parallel}^\top \hat{\mathbf{e}}_{i,s} & \hat{\mathbf{e}}_{i,\parallel}^\top \hat{\mathbf{e}}_{i,p} \end{bmatrix} \begin{bmatrix} E_{i,s} \\ E_{i,p} \end{bmatrix} = \mathbf{W}(\hat{\mathbf{e}}_{i,\perp}, \hat{\mathbf{e}}_{i,\parallel}, \hat{\mathbf{e}}_{i,s}, \hat{\mathbf{e}}_{i,p}) \begin{bmatrix} E_{i,s} \\ E_{i,p} \end{bmatrix} \end{aligned} \quad (104)$$

where we have defined the following matrix-valued function

$$\mathbf{W}(\hat{\mathbf{a}}, \hat{\mathbf{b}}, \hat{\mathbf{q}}, \hat{\mathbf{r}}) = \begin{bmatrix} \hat{\mathbf{a}}^\top \hat{\mathbf{q}} & \hat{\mathbf{a}}^\top \hat{\mathbf{r}} \\ \hat{\mathbf{b}}^\top \hat{\mathbf{q}} & \hat{\mathbf{b}}^\top \hat{\mathbf{r}} \end{bmatrix}. \quad (105)$$

While the angles of incidence and reflection are both equal to θ_1 , the angle of the refracted wave θ_2 is given by Snell's law

$$\sin(\theta_2) = \sqrt{\frac{\eta_1}{\eta_2}} \sin(\theta_1) \quad (106)$$

or, equivalently,

$$\cos(\theta_2) = \sqrt{1 - \frac{\eta_1}{\eta_2} \sin^2(\theta_1)}. \quad (107)$$

The reflected and transmitted wave phasors \mathbf{E}_r and \mathbf{E}_t are similarly represented as

$$\mathbf{E}_r = E_{r,\perp} \hat{\mathbf{e}}_{r,\perp} + E_{r,\parallel} \hat{\mathbf{e}}_{r,\parallel} \quad (108)$$

$$\mathbf{E}_t = E_{t,\perp} \hat{\mathbf{e}}_{t,\perp} + E_{t,\parallel} \hat{\mathbf{e}}_{t,\parallel} \quad (109)$$

where

$$\begin{aligned}\hat{\mathbf{e}}_{r,\perp} &= \hat{\mathbf{e}}_{i,\perp} \\ \hat{\mathbf{e}}_{r,\parallel} &= \frac{\hat{\mathbf{e}}_{r,\perp} \times \hat{\mathbf{k}}_r}{\left\| \hat{\mathbf{e}}_{r,\perp} \times \hat{\mathbf{k}}_r \right\|_2} \\ \hat{\mathbf{e}}_{t,\perp} &= \hat{\mathbf{e}}_{i,\perp} \\ \hat{\mathbf{e}}_{t,\parallel} &= \frac{\hat{\mathbf{e}}_{t,\perp} \times \hat{\mathbf{k}}_t}{\left\| \hat{\mathbf{e}}_{t,\perp} \times \hat{\mathbf{k}}_t \right\|_2}\end{aligned}\quad (110)$$

and

$$\begin{aligned}\hat{\mathbf{k}}_r &= \hat{\mathbf{k}}_i - 2 \left(\hat{\mathbf{k}}_i^\top \hat{\mathbf{n}} \right) \hat{\mathbf{n}} \\ \hat{\mathbf{k}}_t &= \sqrt{\frac{\eta_1}{\eta_2}} \hat{\mathbf{k}}_i + \left(\sqrt{\frac{\eta_1}{\eta_2}} \cos(\theta_1) - \cos(\theta_2) \right) \hat{\mathbf{n}}.\end{aligned}\quad (111)$$

The *Fresnel* equations provide relationships between the incident, reflected, and refracted field components for $\sqrt{|\eta_1/\eta_2|} \sin(\theta_1) < 1$:

$$\begin{aligned}r_\perp &= \frac{E_{r,\perp}}{E_{i,\perp}} = \frac{\sqrt{\eta_1} \cos(\theta_1) - \sqrt{\eta_2} \cos(\theta_2)}{\sqrt{\eta_1} \cos(\theta_1) + \sqrt{\eta_2} \cos(\theta_2)} \\ r_\parallel &= \frac{E_{r,\parallel}}{E_{i,\parallel}} = \frac{\sqrt{\eta_2} \cos(\theta_1) - \sqrt{\eta_1} \cos(\theta_2)}{\sqrt{\eta_2} \cos(\theta_1) + \sqrt{\eta_1} \cos(\theta_2)} \\ t_\perp &= \frac{E_{t,\perp}}{E_{i,\perp}} = \frac{2\sqrt{\eta_1} \cos(\theta_1)}{\sqrt{\eta_1} \cos(\theta_1) + \sqrt{\eta_2} \cos(\theta_2)} \\ t_\parallel &= \frac{E_{t,\parallel}}{E_{i,\parallel}} = \frac{2\sqrt{\eta_1} \cos(\theta_1)}{\sqrt{\eta_2} \cos(\theta_1) + \sqrt{\eta_1} \cos(\theta_2)}.\end{aligned}\quad (112)$$

If $\sqrt{|\eta_1/\eta_2|} \sin(\theta_1) \geq 1$, we have $r_\perp = r_\parallel = 1$ and $t_\perp = t_\parallel = 0$, i.e. total reflection. For the case of an incident wave in vacuum, i.e. $\eta_1 = 1$, the Fresnel equations (112) simplify to

$$\begin{aligned}r_\perp &= \frac{\cos(\theta_1) - \sqrt{\eta_2 - \sin^2(\theta_1)}}{\cos(\theta_1) + \sqrt{\eta_2 - \sin^2(\theta_1)}} \\ r_\parallel &= \frac{\eta_2 \cos(\theta_1) - \sqrt{\eta_2 - \sin^2(\theta_1)}}{\eta_2 \cos(\theta_1) + \sqrt{\eta_2 - \sin^2(\theta_1)}} \\ t_\perp &= \frac{2 \cos(\theta_1)}{\cos(\theta_1) + \sqrt{\eta_2 - \sin^2(\theta_1)}} \\ t_\parallel &= \frac{2\sqrt{\eta_2} \cos(\theta_1)}{\eta_2 \cos(\theta_1) + \sqrt{\eta_2 - \sin^2(\theta_1)}}.\end{aligned}\quad (113)$$

Putting everything together, we obtain the following relationships between incident, reflected, and transmitted waves:

$$\begin{bmatrix} E_{r,\perp} \\ E_{r,\parallel} \end{bmatrix} = \begin{bmatrix} r_\perp & 0 \\ 0 & r_\parallel \end{bmatrix} \mathbf{W} (\hat{\mathbf{e}}_{i,\perp}, \hat{\mathbf{e}}_{i,\parallel}, \hat{\mathbf{e}}_{i,s}, \hat{\mathbf{e}}_{i,p}) \begin{bmatrix} E_{i,s} \\ E_{i,p} \end{bmatrix}\quad (114)$$

$$\begin{bmatrix} E_{t,\perp} \\ E_{t,\parallel} \end{bmatrix} = \begin{bmatrix} t_\perp & 0 \\ 0 & t_\parallel \end{bmatrix} \mathbf{W} (\hat{\mathbf{e}}_{i,\perp}, \hat{\mathbf{e}}_{i,\parallel}, \hat{\mathbf{e}}_{i,s}, \hat{\mathbf{e}}_{i,p}) \begin{bmatrix} E_{i,s} \\ E_{i,p} \end{bmatrix}.\quad (115)$$

A.7.1. Single-Layer Slab

The reflection and refraction coefficients described above assume that the object reflecting the wave or allowing it to penetrate is of infinite size (or thickness). However, since this is rarely the case, it is often more practical

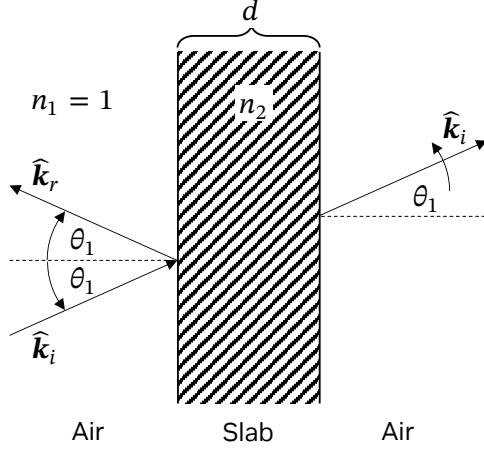


Figure 27: Reflection and refraction of a plane wave at a single-layer slab.

to assume that the object has a finite thickness. In such cases, the object can be modeled as a slab consisting of a single layer made of the same material, as shown in Figure 27. The reflection and transmission coefficients, which should be used instead of (113), are then computed as described in [25, Section 2.2.2.2]:

$$r = \frac{r' (1 - e^{-j2q})}{1 - r'^2 e^{-j2q}} \quad (116)$$

$$t = \frac{(1 - r'^2) e^{-jq}}{1 - r'^2 e^{-j2q}} \quad (117)$$

where

$$q = \frac{2\pi d}{\lambda} \sqrt{\eta - \sin^2 \theta_0} \quad (118)$$

d [m] is the thickness of the slab, η the complex relative permittivity as defined in (65), and r' denotes either r_\perp or r_\parallel from (113), depending on the polarization of the incident electric field.

A.8. Diffuse Reflection

When an electromagnetic wave impinges on a surface, one part of the energy gets reflected while the other part gets refracted, i.e. it propagates into the surface. We distinguish between two types of reflection, specular and diffuse. The former type is discussed in Section A.7 and we will focus now on the latter type. When a rays hits a diffuse reflection surface, it is not reflected into a single (specular) direction but rather scattered toward many different directions. Since most surfaces give both specular and diffuse reflections, we denote by S^2 the fraction of the reflected energy that is diffusely scattered, where $S \in [0, 1]$ is the so-called *scattering coefficient* [26]. Similarly, R^2 is the specularly reflected fraction of the reflected energy, where $R \in [0, 1]$ is the *reflection reduction factor*. The following relationship between R and S holds:

$$R = \sqrt{1 - S^2}. \quad (119)$$

Whenever a material has a scattering coefficient $S > 0$, the Fresnel reflection coefficients in (112) must be multiplied by (119).

Let us consider an incoming locally planar linearly polarized wave with field phasor $\mathbf{E}_i(\mathbf{q})$ at the scattering point \mathbf{q} on the surface, as shown in Figure 28. We focus on the scattered field of an infinitesimally small surface element dA in the direction $\hat{\mathbf{k}}_s$. Note that the surface normal $\hat{\mathbf{n}}$ has an arbitrary orientation with respect to the global coordinate system, whose (x, y, z) axes are shown in green dotted lines. Also, the small surface element is related to the incident ray tube solid angle $d\omega$ through $dA = r^2 d\omega$, where r is the ray tube length.

The incoming field phasor can be represented by two arbitrary orthogonal polarization components (both

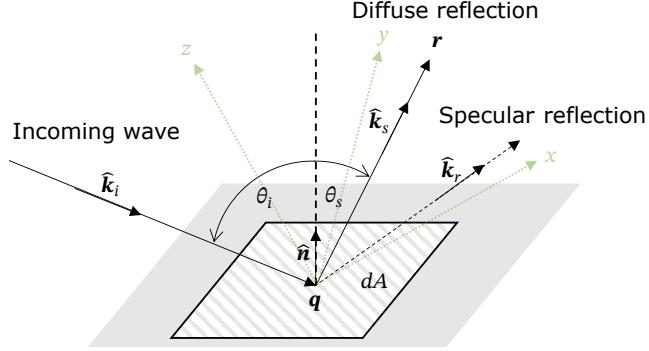


Figure 28: Diffuse and specular reflection of an incoming wave.

orthogonal to the incoming wave vector $\hat{\mathbf{k}}_i$):

$$\mathbf{E}_i(\mathbf{q}) = E_{i,s}\hat{\mathbf{e}}_{i,s} + E_{i,p}\hat{\mathbf{e}}_{i,p} \quad (120)$$

$$= E_{i,\perp}\hat{\mathbf{e}}_{i,\perp} + E_{i,\parallel}\hat{\mathbf{e}}_{i,\parallel} \quad (121)$$

$$= E_{i,\theta}\hat{\boldsymbol{\theta}}(\hat{\mathbf{k}}_i) + E_{i,\phi}\hat{\boldsymbol{\varphi}}(\hat{\mathbf{k}}_i) \quad (122)$$

where we have omitted the dependence of the field strength on the position \mathbf{q} for brevity. The second representation via $(E_{i,\perp}, E_{i,\parallel})$ is used for the computation of the specularly reflected field as explained in Section A.7. The third representation decomposes the field into a vertically and horizontally polarized component, where $\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\varphi}}$ are defined in (49).

According to [27, Eq. 9], the diffusely scattered field $\mathbf{E}_s(\mathbf{r})$ at the observation point \mathbf{r} can be modeled as $\mathbf{E}_s(\mathbf{r}) = E_{s,\theta}\hat{\boldsymbol{\theta}}(\hat{\mathbf{k}}_s) + E_{s,\varphi}\hat{\boldsymbol{\varphi}}(\hat{\mathbf{k}}_s)$, where the orthogonal field components are computed as

$$\begin{bmatrix} E_{s,\theta} \\ E_{s,\varphi} \end{bmatrix} = \frac{S\Gamma}{\|\mathbf{r} - \mathbf{q}\|_2} \sqrt{f_s(\hat{\mathbf{k}}_i, \hat{\mathbf{k}}_s, \hat{\mathbf{n}})} \cos(\theta_i) dA \begin{bmatrix} \sqrt{1 - K_x} e^{j\chi_1} & -\sqrt{K_x} e^{j\chi_1} \\ \sqrt{K_x} e^{j\chi_2} & \sqrt{1 - K_x} e^{j\chi_2} \end{bmatrix} \begin{bmatrix} E_{i,\theta} \\ E_{i,\varphi} \end{bmatrix}. \quad (123)$$

Here, Γ^2 is the percentage of the incoming power that is reflected (specularly and diffuse), which can be computed as

$$\Gamma = \frac{\sqrt{|r_\perp E_{i,\perp}|^2 + |r_\parallel E_{i,\parallel}|^2}}{\|\mathbf{E}_i(\mathbf{q})\|_2} \quad (124)$$

where r_\perp, r_\parallel are defined in (112), $\chi_1, \chi_2 \in [0, 2\pi]$ are (optional) independent random phase shifts, and the quantity $K_x \in [0, 1]$ is defined by the scattering cross-polarization discrimination

$$\text{XPD}_s = 10 \log_{10} \left(\frac{|E_{s,\text{pol}}|^2}{|E_{s,\text{xpol}}|^2} \right) = 10 \log_{10} \left(\frac{1 - K_x}{K_x} \right). \quad (125)$$

This quantity determines how much energy gets transferred from one polarization direction into the other through the scattering process. Lastly, dA is the size of the small area element on the reflecting surface under consideration, and $f_s(\hat{\mathbf{k}}_i, \hat{\mathbf{k}}_s, \hat{\mathbf{n}})$ is the *scattering pattern*, which has similarities with the **BSDF** in computer graphics [7, Ch. 4.3.1]. The scattering pattern must be normalized to satisfy the condition

$$\int_0^{\pi/2} \int_0^{2\pi} f_s(\hat{\mathbf{k}}_i, \hat{\mathbf{k}}_s, \hat{\mathbf{n}}) \sin(\theta_s) d\phi_s d\theta_s = 1 \quad (126)$$

which ensures the power balance between the incoming, reflected, and refracted fields.

Example scattering patterns

The authors of [26] derived several simple scattering patterns that were shown to achieve good agreement with measurements when correctly parametrized.

Lambertian: This model describes a perfectly diffuse scattering surface whose *scattering radiation lobe* has its maximum in the direction of the surface normal:

$$f_s^{\text{Lambert}}(\hat{\mathbf{k}}_i, \hat{\mathbf{k}}_s, \hat{\mathbf{n}}) = \frac{\hat{\mathbf{n}}^\top \hat{\mathbf{k}}_s}{\pi} = \frac{\cos(\theta_s)}{\pi} \quad (127)$$

Directive: This model assumes that the scattered field is concentrated around the direction of the specular reflection $\hat{\mathbf{k}}_r$ (defined in (111)). The width of the scattering lobe can be controlled via the integer parameter $\alpha_R = 1, 2, \dots$:

$$\begin{aligned} f_s^{\text{directive}}(\hat{\mathbf{k}}_i, \hat{\mathbf{k}}_s, \hat{\mathbf{n}}) &= F_{\alpha_R}(\theta_i)^{-1} \left(\frac{1 + \hat{\mathbf{k}}_r^\top \hat{\mathbf{k}}_s}{2} \right)^{\alpha_R} \\ F_\alpha(\theta_i) &= \frac{1}{2^\alpha} \sum_{k=0}^{\alpha} \binom{\alpha}{k} I_k, \quad \theta_i = \cos^{-1}(-\hat{\mathbf{k}}_i^\top \hat{\mathbf{n}}) \\ I_k &= \frac{2\pi}{k+1} \begin{cases} 1 & k \text{ even} \\ \cos(\theta_i) \sum_{w=0}^{(k-1)/2} \binom{2w}{w} \frac{\sin^{2w}(\theta_i)}{2^{2w}} & k \text{ odd} \end{cases} \end{aligned} \quad (128)$$

Backscattering: This model adds a scattering lobe to the directive model described above which points toward the direction from which the incident wave arrives (i.e. $-\hat{\mathbf{k}}_i$). The width of this lobe is controlled by the parameter $\alpha_I = 1, 2, \dots$. The parameter $\Lambda \in [0, 1]$ determines the distribution of energy between both lobes. For $\Lambda = 1$, this models reduces to the directive model.

$$\begin{aligned} f_s^{\text{bs}}(\hat{\mathbf{k}}_i, \hat{\mathbf{k}}_s, \hat{\mathbf{n}}) &= F_{\alpha_R, \alpha_I}(\theta_i)^{-1} \left[\Lambda \left(\frac{1 + \hat{\mathbf{k}}_r^\top \hat{\mathbf{k}}_s}{2} \right)^{\alpha_R} + (1 - \Lambda) \left(\frac{1 - \hat{\mathbf{k}}_i^\top \hat{\mathbf{k}}_s}{2} \right)^{\alpha_I} \right] \\ F_{\alpha, \beta}(\theta_i)^{-1} &= \Lambda F_\alpha(\theta_i) + (1 - \Lambda) F_\beta(\theta_i) \end{aligned} \quad (129)$$

References

- [1] J. Hoydis, F. Aït Aoudia, S. Cammerer, M. Nimier-David, N. Binder, G. Marcus, and A. Keller, “Sionna RT: Differentiable ray tracing for radio propagation modeling,” *arXiv:2303.11103*, 2023. [Online]. Available: <https://arxiv.org/abs/2303.11103> 4
- [2] J. Hoydis, S. Cammerer, F. Aït Aoudia, A. Vem, N. Binder, G. Marcus, and A. Keller, “Sionna: An open-source library for next-generation physical layer research,” *arXiv:2203.11854*, 2022. [Online]. Available: <https://arxiv.org/abs/2203.11854> 4
- [3] W. Jakob, S. Speierer, N. Roussel, and D. Vicini, “Dr.Jit: A just-in-time compiler for differentiable rendering,” *Transactions on Graphics (Proceedings of SIGGRAPH)*, vol. 41, no. 4, Jul. 2022. 4
- [4] W. Jakob, S. Speierer, N. Roussel, M. Nimier-David, D. Vicini, T. Zeltner, B. Nicolet, M. Crespo, V. Leroy, and Z. Zhang, “Mitsuba 3 Physically Based Renderer,” 2022. [Online]. Available: <https://mitsuba-renderer.org> 4
- [5] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015. [Online]. Available: <https://www.tensorflow.org/> 4
- [6] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köf, E. Yang, Z. DeVito, M. Raison, , A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “PyTorch: An imperative style, high-performance deep learning library,” 2019. [Online]. Available: <https://pytorch.org> 4
- [7] M. Pharr, W. Jakob, and G. Humphreys, *Physically Based Rendering: From Theory to Implementation*, 4th ed. The MIT Press, 2023. 5, 6, 41
- [8] OpenStreetMap, “Planet dump retrieved from <https://planet.osm.org> ,” <https://www.openstreetmap.org>, 2017. 6
- [9] Prochitecture, “Blosm for Blender: OpenStreetMap, Google 3D cities, terrain,” accessed 18-February-2025. [Online]. Available: <https://github.com/vvoovv/blosm> 6
- [10] Wikipedia. Diffuse reflection. [Online]. Available: https://en.wikipedia.org/wiki/Diffuse_reflection 8
- [11] D. A. McNamara, C. W. I. Pistorius, and J. Malherbe, *Introduction to the Uniform Geometrical Theory of Diffraction*. Artech House, 1990. 8
- [12] Á. González, “Measurement of areas on a sphere using Fibonacci and latitude–longitude lattices,” *Mathematical Geosciences*, vol. 42, pp. 49–64, 2010. 14
- [13] J. Hannay and J. Nye, “Fibonacci numerical integration on a sphere,” *Journal of Physics A: Mathematical and General*, vol. 37, no. 48, p. 11591, 2004. 14
- [14] Wikipedia. Importance sampling. Accessed 18-February-2025. [Online]. Available: https://en.wikipedia.org/wiki/Importance_sampling 15
- [15] ——. Pairing function. Accessed 18-February-2025. [Online]. Available: https://en.wikipedia.org/wiki/Pairing_function#Cantor_pairing_function 16
- [16] ——. Rolling hash. [Online]. Available: https://en.wikipedia.org/wiki/Rolling_hash 16
- [17] 3rd Generation Partnership Project (3GPP), “Study on channel model for frequencies from 0.5 to 100 GHz,” 3GPP, Technical Report TR 38.901, 2024, Release 18.0.0. [Online]. Available: <https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3173> 18, 32

- [18] F. Aït Aoudia, J. Hoydis, M. Nimier-David, S. Cammerer, and A. Keller, “Instant radio maps,” 2024. [Online]. Available: <https://github.com/NVlabs/instant-rm> 31
- [19] N. Geng and W. Wiesbeck, *Planungsmethoden für die Mobilkommunikation: Funknetzplanung unter realen physikalischen Ausbreitungsbedingungen*. Springer-Verlag, 2013. 32
- [20] Wikipedia. atan2. Accessed 10-February-2025. [Online]. Available: <https://en.wikipedia.org/wiki/Atan2> 32
- [21] ——. Rodrigues’ rotation formula. Accessed 10-February-2025. [Online]. Available: https://en.wikipedia.org/wiki/Rodrigues%27_rotation_formula 33
- [22] C. A. Balanis, *Advanced Engineering Electromagnetics*. John Wiley & Sons, 2012. 33
- [23] Wikipedia. Maximum power transfer theorem. Accessed 10-February-2025. [Online]. Available: https://en.wikipedia.org/wiki/Maximum_power_transfer_theorem 35
- [24] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge University Press, 2005. [Online]. Available: https://web.stanford.edu/~dntse/wireless_book.html 37
- [25] ITU, “Recommendation ITU-R P.2040-3: Effects of building materials and structures on radiowave propagation above about 100 MHz,” International Telecommunication Union, Recommendation ITU-R P.2040-3, August 2023. [Online]. Available: <https://www.itu.int/rec/R-REC-P.2040/en> 37, 40
- [26] V. Degli-Esposti, F. Fuschini, E. M. Vitucci, and G. Falciai, “Measurement and modelling of scattering from buildings,” *IEEE Transactions on Antennas and Propagation*, vol. 55, no. 1, pp. 143–153, 2007. 40, 42
- [27] V. Degli-Esposti, V.-M. Kolmonen, E. M. Vitucci, and P. Vainikainen, “Analysis and modeling on co- and cross-polarized urban radio propagation for dual-polarized MIMO wireless systems,” *IEEE Transactions on Antennas and Propagation*, vol. 59, no. 11, pp. 4247–4256, 2011. 41