

Accelerated 3-D Multi-Diffraction Image-Theory Ray Tracing for 6G Propagation with an Urban-Scale Sionna RT Benchmark

Hyunjun Yang, *Student Member, IEEE*, Hooyoung Kim, *Student Member, IEEE*, Jungsuek Oh, *Senior Member, IEEE*

Abstract—We present an image-theory (IT)-based, CPU/GPU-accelerated full 3-D ray-tracing (RT) framework that integrates multi-diffraction (MD) for arbitrary urban wedge configurations in 6G upper-mid-band scenarios. The framework validates the UTD MD path-tracing and field-evaluation blocks and shows that incorporating second-order UTD removes the transition-region discontinuities observed in a single-diffraction-only engine. To handle singularity-sensitive double-diffraction paths near the $Q_1 \approx Q_2$ collapse region, we introduce a singularity-guarded Newton (SGN) optimizer that preserves the fast Carluccio-Albani (C-A) update path for regular candidates while selectively activating guarded stabilization. For efficient GPU execution, the proposed device-compacted split SGN confines stabilized correction to guard-active paths, reaching approximately 42.9 Mpaths/s and retaining about 86.5–86.6% of the C-A GPU baseline throughput in the saturation regime. At the full-engine level, we combine multi-diffraction anxel-beam shrinkage (DABS)-based visibility-tree construction, NVIDIA OptiX-based path tracing/shadowing, and a compact valid-path pipeline. In a measurement-grounded urban CAD scenario, the final $R = 6$, $D = 2$ compact-pipeline simulation computes 1,021,242 total paths, including 635,823 MD paths, in 2693.4 s with 22.9 GB peak GPU memory, enabling a full 3-D MD IT-RT simulation scale that was previously impractical for conventional IT-based RT engines. Under the same Tx/Rx layout, this path set is more than ten times larger than the 94,581 rays identified by Sionna ray-depth 5 and edge diffraction enabled, while the added cost of enabling double diffraction over the $D = 1$ case remains only 130.4 s, or 5.1%. The results demonstrate that robust full-3-D MD-aware IT-RT provides a more complete multipath basis for 6G upper-mid-band coverage analysis and practical RT-simulator development.

Index Terms—6G Upper Mid-band, Multi-Diffraction, Image Theory, Full 3D Ray Tracing, Urban Scale Simulation, CPU and GPU parallel computation.

I. INTRODUCTION

AS 6G deployment studies move toward wider bandwidth and reliable outdoor coverage, the upper-mid band (7–24 GHz) has become an important candidate spectrum range

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (RS-2025-02283217, Development of AI-based Frequency Interference Analysis and Electromagnetic Wave Prediction Technology, 50%), and (RS-2019-0-00098, Advanced and Integrated Software Development for Electromagnetic Analysis, 50%). (*Corresponding author: Jungsuek Oh*).

Hyunjun Yang, Hooyoung Kim and Jungsuek Oh are with the Institute of New Media and Communications (INMC) and the Department of Electrical and Computer Engineering, Seoul National University, Seoul, South Korea (e-mails: guswns6217@snu.ac.kr, and jungsuek@snu.ac.kr)

between current sub-6 GHz allocations and mmWave bands (>24 GHz). At the World Radiocommunication Conference (WRC)-23, the 4.4–4.8 GHz, 7.125–8.4 GHz, and 14.8–15.35 GHz were highlighted as prime International Mobile Telecommunications (IMT)-2030 candidates, and the International Telecommunication Union (ITU) tasked each region with feasibility studies before WRC-27. In particular, the 7.125–8.4 GHz range is drawing interest for 6G outdoor deployment because of its favorable coverage [1]. At 7.5 GHz—within ITU Region 3’s 7.125–8.4 GHz study band—recent dense-urban and urban campaigns have used measurement-calibrated ray tracing (RT) to analyze the additional gain required to approach sub-6-GHz-like coverage [2], [3]. Building on these measurement-grounded studies, the present work focuses on explicit multi-diffraction (MD) path generation and its computational integration into a full three-dimensional (3-D) image-theory RT (IT-RT) engine. This enables a dedicated study of higher-order diffraction for refining upper-mid-band coverage prediction.

MD matters because single-edge uniform theory of diffraction (UTD) cannot capture scenarios where a singly diffracted ray is itself shadowed or where the source/observer lies in overlapping transition regions; in such regimes, the second-order term can rival the first, and higher orders may be non-negligible in rich outdoor multipath [4]. This limitation is especially consequential in the upper-mid band, where diffraction contributes more to non-line-of-sight (NLOS) coverage and cell-edge reliability than at mmWave (>24 GHz) [3].

The electromagnetic theory for MD has advanced from closed-form solutions for parallel wedges [5], [6], to coplanar skew-wedge formulations using generalized Fresnel integrals (GFI) that ensure continuity for both polarizations [4], and ultimately to a UTD transition-function (T) framework that uniformly handles arbitrary 3-D wedge configurations [7]. Building on this foundation, a dyadic second-order double-diffraction coefficient for wedges with general boundaries was derived as a cascade of single-wedge geometrical theory of diffraction (GTD) dyadics multiplied by T ; it reduces to the perfect electric conductor (PEC) limit and remains uniform across overlapping transitions [8].

Despite this theoretical maturity, practical urban-scale RT engines still face substantial barriers to full-3-D MD integration. Many recent RT engines either omit MD paths [9]–[13] or restrict diffraction modeling to partially two-dimensional (2-D) or parallel-wedge configurations [14]. This is because

MD integration requires more than a field coefficient: it also requires robust evaluation of special numerical functions such as T functions [15] and GFI [16], visibility-tree (VT) construction for large numbers of wedge pairs, and MD path finding in arbitrary 3-D geometries through nonlinear optimization [17].

RT techniques are commonly categorized into shooting-and-bouncing rays (SBR) and IT. SBR offers faster run-times than IT and is widely adopted in RT simulations; however, persistent issues with cutoff, missing paths, and overcounting remain [9], [10], [12], [18]. Several SBR studies have improved computational performance through parallel computing, but diffracted rays were omitted from the validation scenarios [9], [10], [12]. IT is widely regarded as a high-fidelity reference for developing and validating SBR algorithms because it computes geometrically valid ray contributions without omission or overcounting [10]; however, IT naturally requires constructing a facets-and-wedges VT, which is time-consuming and frequently becomes the main bottleneck. Within IT-RT, acceleration has been pursued using angular Z-buffer (AZB) methods [19]–[21] and illumination-zone methods [22], [23], but these approaches can be difficult to parallelize or may simplify the geometry by restricting the domain to 2-D structures or separately treating vertical-vertical (V-V) and horizontal-horizontal (H-H) planes. Such restrictions conflict with the goal of urban-scale MD in IT-RT over arbitrary 3-D wedge configurations.

Commercial and open-source RT tools also show that robust MD support remains challenging in practice. Previous studies [24], [25] and technical reports or release notes [26], [27] indicate that Wireless InSite is documented to support up to three diffractions [25], but benchmarking studies have effectively reported results only for single-diffraction paths [24]. The MathWorks Ray Tracing Model supports only first-order diffraction in the reported benchmark setting [24], while NVIDIA Sionna RT [26] and FEKO WinProp (IT-RT) [27] have undergone progressive updates related to diffraction, including double diffraction (DD). Similarly, many upper-mid-band and mmWave RT studies still focus on reflection, line-of-sight propagation, or at most single diffraction [2], [11], [14], [28], [29]. Although our previous IT-RT engines were accelerated and validated using urban measurements [2], [28], the detailed development, stabilization, acceleration, and path-completeness analysis of arbitrary full-3-D MD were outside their main scope.

Recent 6G studies have also used ray tracing for applications beyond coverage prediction, including integrated sensing and communication (ISAC) imaging. For example, [30] proposed a 6.75-GHz framework that maps channel-state-information-derived multipath components to equivalent reflection points for 3-D object reconstruction. Such studies highlight the value of RT for NLOS and multipath-rich sensing applications, where many propagation paths must be generated with both physical completeness and computational efficiency. This further motivates a robust and accelerated full-3-D MD IT-RT engine for complex urban environments.

Consequently, although GPU/CPU parallelization has accelerated many RT pipelines, robust and scalable MD over arbitrary 3-D wedge configurations remains limited. Beyond

the UTD MD field formulation, a key remaining challenge is robust path optimization for singularity-sensitive wedge configurations in large urban CAD scenarios. With 6G moving toward the upper-mid band, where diffraction-assisted propagation has a larger impact than at mmWave, closing this gap is increasingly important.

This paper presents a scalable, full-3-D MD ray-tracing engine based on image theory for arbitrary urban wedge configurations. The proposed framework validates the UTD MD path-tracing and field-evaluation blocks, introduces a singularity-guarded Newton (SGN) optimizer for robust double-diffraction path recovery, and reorganizes the guarded optimizer for GPU-efficient execution. It further integrates the resulting solver into a CPU/GPU-accelerated IT-RT pipeline with MD anisotropic beam shrinkage (DABS)-based VTC and compact valid-path processing. Finally, by comparing the proposed engine with Sionna RT in a large urban CAD scenario, we show that the proposed robust full-3-D IT-based RT approach provides a more complete multipath basis for 6G upper-mid-band coverage and practical RT-simulator development.

II. BASELINE MULTI-DIFFRACTION RAY SOLVER

This section organizes the MD ray formulation for arbitrary 3-D wedges into path tracing and field calculation, and implements these theoretical models in C++. We build on the UTD multi-diffraction framework for arbitrary 3-D wedges by Albani et al., Carluccio et al., and Puggelli et al. [7], [8], [15]–[17], which guided our C++ implementation, and we validate it against their reference models. Validation is conducted on Albani's numerical examples, verifying Keller's law and comparing against MATLAB numerical integration (NI) results to confirm accuracy. Furthermore, we compare the newly implemented MD-enabled RT simulator with our previous single-diffraction (SD)-only simulator [2], [28] and show that it accurately captures field propagation—including transition regions—and MD paths that the prior engine could not resolve. These results indicate that the proposed MD implementation will improve propagation prediction by comprehensively accounting for the diffraction propagation characteristics in the 6G upper mid-band [3].

A. Multi-Diffraction Path Tracing

We consider a transmitter (Tx) at P' and a receiver (Rx) at P , and two wedge edges $\mathcal{E}_1$ and $\mathcal{E}_2$. Each edge is represented by a point-direction pair $(O_i, \hat{\zeta}_i)$, $i \in \{1, 2\}$, so a generic diffraction point on edge i is

$$Q_i(\zeta_i) = O_i + \zeta_i \hat{\zeta}_i, \quad \zeta_i \in \mathbb{R}. \quad (1)$$

Here, O_i is any fixed point on edge $\mathcal{E}_i$ (e.g., an endpoint), and $\hat{\zeta}_i$ is a unit vector collinear with the edge; the scalar ζ_i is the signed arclength along $\hat{\zeta}_i$ so that $Q_i(\zeta_i) = O_i + \zeta_i \hat{\zeta}_i$. For a finite edge with endpoints A_i and B_i , one may set $O_i = A_i$, $\hat{\zeta}_i = (B_i - A_i)/\|B_i - A_i\|$, and restrict $\zeta_i \in [0, \|B_i - A_i\|]$, which makes the segment test and the differentiation of $L(\zeta)$ straightforward.

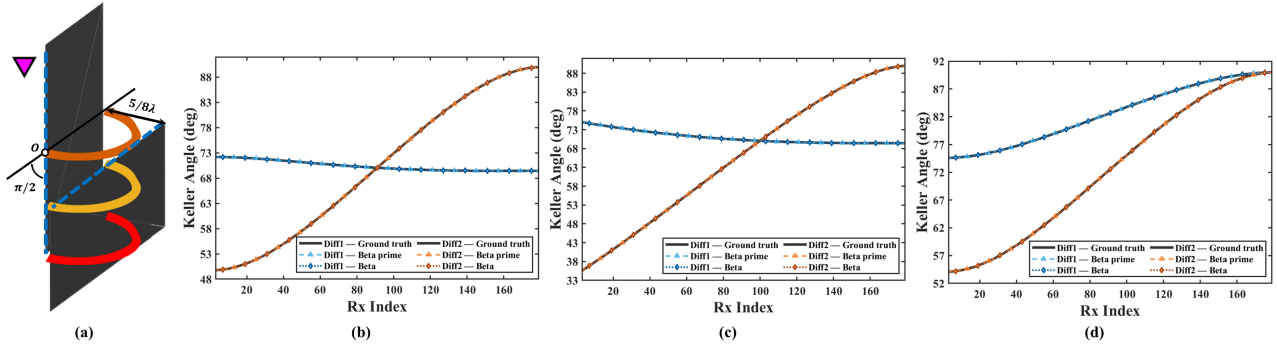


Fig. 1. Implementation of the MD path-tracing block for the validation geometry with an edge angle of $\psi = \pi/2$. The wedge parameters are $(A_1, B_1, \hat{\zeta}_1) = ((0, 0, 1), (0, 0, -1), (0, 1, 0))$ and $(A_2, B_2, \hat{\zeta}_2) = ((0.625, 1, 0), (0.625, -1, 0), (0, 0, -1))$, where A_i and B_i are given in wavelength units. For this geometry, MD paths are computed toward Rx points sampled on three semicircular arcs; the incident/diffracted angles at both wedges satisfy Keller's law ($\beta'_1 = \beta_1, \beta'_2 = \beta_2$), confirming the physical validity of the traced paths.

By Fermat's principle, the minimum of the following cost function $L(\zeta)$ equals the DD path length, and the corresponding Q_1 and Q_2 are the diffraction points:

$$L(\zeta) = \|Q_1 - P'\| + \|Q_2 - Q_1\| + \|P - Q_2\|, \quad \zeta = [\zeta_1, \zeta_2]^T. \quad (2)$$

Fig. 1 shows our code-based implementation of the MD path-tracing block for the validation geometry with an edge angle of $\psi = \pi/2$. This geometry follows the arbitrary 3-D wedge configuration used for high-order UTD analysis in [7]. Two half-planes with $n_1 = n_2 = 2$ are placed on parallel planes separated by $5\lambda/8$, with the second half-plane located between the source and the first. Each half-plane is modeled as a $2\lambda \times 1\lambda$ rectangular plate, and the detailed wedge parameters are specified in the caption of Fig. 1. The Tx is located at $(x, y, z) = (-0.5, 0.5, 0.5)\lambda$, and the Rx points are sampled on three semicircular arcs of radius $\lambda/2$ centered at $(x, y) = (5/8, -0.5)\lambda$, with the arcs located in the planes $z \in \{-\lambda/2, 0, \lambda/2\}$. The angle α parameterizes each arc over 0° – 180° and maps one-to-one to the Rx index used in the Keller-angle plots.

For this geometry, we compute the DD path from the Tx to every Rx point on the three semicircular arcs. For both the first and second wedges, the computed incident and diffracted angles satisfy Keller's law, i.e., $\beta'_1 = \beta_1$ and $\beta'_2 = \beta_2$. This confirms the physical validity of the traced paths and verifies the accuracy of the implemented path-search algorithm for the considered arbitrary 3-D wedge configuration.

B. Evaluation of the UTD Transition Function $T(a, b, w)$

SD is treated with a single UTD wedge coefficient and a Fresnel-type transition to maintain shadow-boundary uniformity, whose numerical core reduces to the following standard Fresnel-type integral:

$$F_0(x) = 2j\sqrt{x}e^{jx} \int_{\sqrt{x}}^{\infty} e^{-jt^2} dt. \quad (3)$$

Eq. (3) is a Fresnel-type integral widely used in high-frequency approximation methods, including the geometrical theory of diffraction (GTD), the uniform theory of diffraction (UTD), and the physical theory of diffraction (PTD).

By contrast, in the DD case, simply cascading the SD coefficients from each wedge based on (3) fails to ensure shadow boundary uniformity. For example, in single-transition regions, SD rays abruptly appear or disappear as the observation point crosses the second edge's shadow boundary; in double-transition regions, geometrical-optics (GO) and SD rays simultaneously appear or vanish. Such situations arise frequently in dense urban environments where RT models buildings as rectangular prisms.

Consequently, a uniform treatment based on the canonical three-argument UTD transition function $T(a, b, w)$, together with its generalized Fresnel integral (GFI) machinery, has been developed.

$$\mathcal{T}(a, b, w) = \frac{4\pi j ab}{\sqrt{1-w^2}} \left[\mathcal{G}(a, \bar{a}) + \mathcal{G}(b, \bar{b}) \right]. \quad (4)$$

Here, $\bar{a} = (b + wa)/\sqrt{1-w^2}$ and $\bar{b} = (a + wb)/\sqrt{1-w^2}$; see [15].

$$\mathcal{G}(p, q) = \frac{q}{2\pi} e^{jp^2} \int_p^{\infty} \frac{e^{-jt^2}}{t^2 + q^2} dt. \quad (5)$$

for $\arg(x) \in (-\frac{3\pi}{4}, \frac{\pi}{4})$, and otherwise by

$$\mathcal{G}(p, q) = -\mathcal{G}(-p, q). \quad (6)$$

In [15] and [16], T and the GFI were formulated to be efficiently and robustly evaluable within modern RT codes, thereby establishing essential requirements for integrating DD into RT engines. As can be seen from [15], [16], T and the GFI can be rewritten in terms of the n th-order UTD Fresnel integrals F_n , defined by

$$F_n(x) = \frac{(2j\sqrt{x})^{n+1}}{n!} e^{jx} \int_{\sqrt{x}}^{\infty} (t - \sqrt{x})^n e^{-jt^2} dt. \quad (7)$$

with $\arg(\sqrt{x}) \in (-\frac{3\pi}{4}, \frac{\pi}{4})$ are the UTD Fresnel integrals of order n , defined in [16]. For an alternative expression that is convenient for series expansions (SE), one can recast F_n in terms of the repeated integrals of the complementary error function ($\text{i}^n \text{erfc}$) as

$$F_n(x) = \frac{\sqrt{\pi}}{2} (2e^{j\pi/4}\sqrt{x})^{n+1} e^{jx} i^n \text{erfc}(e^{j\pi/4}\sqrt{x}). \quad (8)$$

Note that, for $n = 0$, (7) is the standard UTD transition function defined in (3). Building on (7)–(8), $i^n \text{erfc}$ can, in turn, be rewritten in terms of Kummer's confluent hypergeometric function ($M(a, b, z)$) (see [31, Eq. (7.2.12)]), which makes explicit the links between F_n and other classical special functions. In the implementation, the defining oscillatory integral in (7) is not evaluated by direct numerical integration during RT. The zeroth-order term F_0 , corresponding to the conventional UTD Fresnel transition function, is evaluated separately using a standard Fresnel cosine/sine-based routine with regime-dependent approximations, including power-series, rational, and asymptotic forms depending on the argument range.

For $n \geq 1$, F_n is evaluated through the repeated-error-function representation in (8). The repeated-error-function term is further expressed using Kummer's confluent hypergeometric function $M(a, b, z)$. With $u = e^{j\pi/4}\sqrt{x}$, the implemented form is

$$i^n \text{erfc}(u) = e^{-u^2} \left[\frac{M\left(\frac{n+1}{2}, \frac{1}{2}, u^2\right)}{2^n \Gamma\left(\frac{n}{2} + 1\right)} - \frac{u M\left(\frac{n}{2} + 1, \frac{3}{2}, u^2\right)}{2^{n-1} \Gamma\left(\frac{n+1}{2}\right)} \right]. \quad (9)$$

The Kummer function is evaluated using a power-series representation for regular argument ranges and a large-argument representation in the asymptotic regime. Thus, the implemented F_n kernel avoids repeated direct integration of oscillatory integrals and provides an order-by-order evaluation of the retained Fresnel-integral orders required for the UTD double-diffraction transition-function implementation.

The subsequent GFI evaluation follows the domain-partitioned $G_0/G_1/G_2$ regime-separation strategy of Carluccio et al. [16]. Therefore, our implementation follows the same canonical decomposition and regime-separation principle as the efficient T -function and GFI algorithms in [15], [16], respectively, while evaluating the required F_n kernel through the Kummer-based repeated-error-function representation rather than through recursive propagation of F_n .

To assess robustness beyond F_n , we analyzed the relative error of T , used directly in the DD coefficient computation:

$$\begin{aligned} \bar{\mathbf{D}}^{dd} = & \frac{1}{8j\pi k n_1 n_2 \sin \beta_1 \sin \beta_2} \sum_{p_2, q_2=1}^2 \sum_{p_1, q_1=1}^2 \bar{\mathbf{F}}_2^{p_2 q_2} \cdot \bar{\mathbf{F}}_1^{p_1 q_1} \cot\left(\frac{\Phi_1^{p_1 q_1}}{2n_1}\right) \\ & \cdot \cot\left(\frac{\Phi_2^{p_2 q_2}}{2n_2}\right) T(a_1^{p_1 q_1}, a_2^{p_2 q_2}, (-1)^{p_1+q_2} \Omega) \end{aligned} \quad (10)$$

The UTD cotangent multiplicative coefficient $\bar{\mathbf{F}}$ and the input arguments of the transition function, a_1, a_2, Ω , are reported in [8, Eqs. (10)–(14) and Table I]. In [8], the geometry-derived coupling parameter Ω is denoted by w ; here, we use Ω to avoid confusion with the generic third argument w of the UTD transition function.

Finally, the GFI and the three-argument UTD transition function $T(a, b, w)$ are implemented in our C++-based RT engine. The reference $T(a, b, w)$ values were obtained in MATLAB using independent direct numerical quadrature. For the three-argument tuples $T(a, b, w)$, we adopted the test

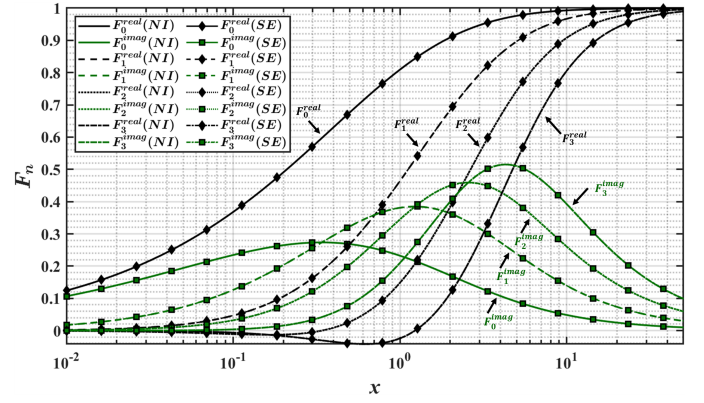


Fig. 2. Verification of the n th-order UTD Fresnel integrals $F_n(x)$ for $n = 0, \dots, 4$: comparison between the proposed C++ implementation and high-accuracy MATLAB numerical-integration references. The tested orders cover the retained F_n range required for the UTD double-diffraction transition-function implementation.

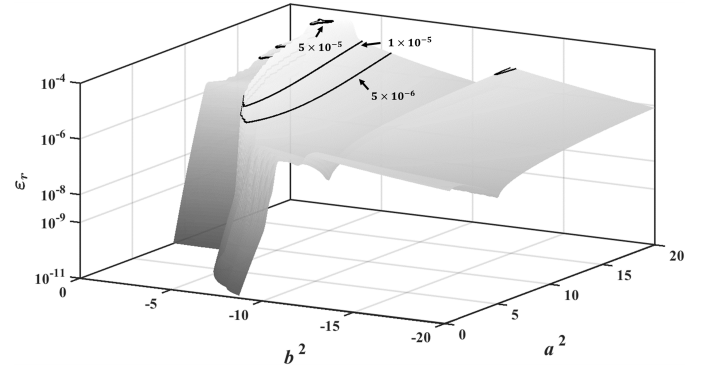


Fig. 3. Relative error of the canonical three-argument UTD transition function $T(a, b, w)$ with the benchmark value $w = 0.5j$ following [15, Fig. 1].

conditions of [15, Fig. 1], where $w = 0.5j$. This case is used as a complex-argument benchmark for the canonical T function. Fig. 3 reports the relative error against the exact T . The maximum relative error is 6×10^{-5} , which is below the 10^{-2} tolerance level used in UTD numerical analysis [32]. The field-level computation was further validated through comparison with a commercial tool [3].

C. Verification of the Base-Solver Implementation

The DD field is expressed, within Albani's ray-fixed coordinate system ([8] and [3, Fig. 3]), as in (11), as the product of the incident field at the first diffraction point ($\mathbf{E}(Q_{D1})$), the DD coefficient ($\bar{\mathbf{D}}^{dd}$), a spreading factor (A^{dd}), and a propagation term.

$$\mathbf{E}^{DD}(s) = \mathbf{E}(Q_{D1}) \cdot \bar{\mathbf{D}}^{dd} A^{dd} e^{-jk(\ell+r_2)} \quad (11)$$

We completed the development of the DD path-finding and coefficient algorithms and evaluated the DD field for arbitrary 3-D wedge configurations. We adopted the dyadic UTD diffraction model of [8], which supports general boundary conditions and applies to arbitrary 3-D wedges. For validation, we used the three PEC wedge cases in Albani's reference geometry [7], corresponding to edge angles of $\psi = \pi/4$,

$\pi/2$, and $3\pi/4$. All coordinates are given in wavelength units. The second wedge is common to all three cases, with $(A_2, B_2, \hat{\zeta}_2) = ((0.625, 1, 0), (0.625, -1, 0), (0, 0, -1))$. The first-wedge parameters are $(A_1, B_1, \hat{\zeta}_1) = ((0, \sqrt{1/2}, \sqrt{1/2}), (0, -\sqrt{1/2}, -\sqrt{1/2}), (0, \sqrt{1/2}, -\sqrt{1/2}))$ for $\psi = \pi/4$, $((0, 0, 1), (0, 0, -1), (0, 1, 0))$ for $\psi = \pi/2$, and $((0, -\sqrt{1/2}, \sqrt{1/2}), (0, \sqrt{1/2}, -\sqrt{1/2}), (0, \sqrt{1/2}, \sqrt{1/2}))$ for $\psi = 3\pi/4$. For a direct comparison with prior work, the Tx placement/pattern and the Rx sampling arc follow [7, Figs. 6–8]: the Tx is a $+\hat{y}$ -oriented Hertzian dipole at $(x, y, z) = (-0.5, 0, 0.001)\lambda$, and the Rx samples on a semicircle in the xz plane centered at $(5/8, 0, 0)\lambda$. All wedges are modeled as perfect electric conductors (PEC).

Fig. 4 compares three results: our previous ray tracer [2], [28], which includes UTD up to first order; the proposed engine, which includes second-order UTD for MD; and Albani's reference solution [7]. The prior engine exhibits field discontinuities near the transition region, whereas these discontinuities are continuously compensated once the DD contribution is included. In addition, the proposed second-order UTD results closely match Albani's model, completing the validation of the DD framework from transition-function evaluation, through path tracing, to field computation.

III. ROBUST MULTI-DIFFRACTION PATH TRACING IN COMPLEX URBAN ENVIRONMENTS

The baseline MD solver described in Section II accurately implements the Carluccio-Albani (C–A) path-search formulation and the associated field-evaluation procedure. For ordinary differentiable double-diffraction configurations, the C–A optimizer converges rapidly and provides physically valid diffraction points. However, complex urban geometries can also generate singularity-sensitive configurations in which two consecutive diffraction points approach each other, i.e. $Q_1 \approx Q_2$. As $\|Q_2 - Q_1\|$ approaches zero, the normalized middle-segment direction in the gradient becomes ill-defined, while the Hessian can become singular or severely ill-conditioned due to its inverse-distance terms [17]. This makes the Newton optimization unstable near the collapse region. This section analyzes this failure mechanism in a representative urban CAD scenario and introduces a singularity-guarded Newton (SGN) stabilization strategy that preserves the fast C–A execution path for regular candidates while adding safeguards only for singularity-sensitive paths.

A. Urban CAD Scenario and Baseline Solver Behavior

To examine the behavior of the MD path optimizer in a practical urban geometry, we used the general urban CAD model reported in [3, Fig. 7]. This model represents the area around Building 71 at Seoul National University (SNU) and comprises 727 facets and 428 wedges over a $345\text{ m} \times 590\text{ m}$ area, as summarized in [3, Table IV]. From this CAD geometry, we generated geometrically feasible wedge pairs and evaluated the double-diffraction path optimization over the corresponding candidate configurations.

To obtain statistically meaningful convergence statistics, we constructed a large diagnostic batch of more than $4 \times$

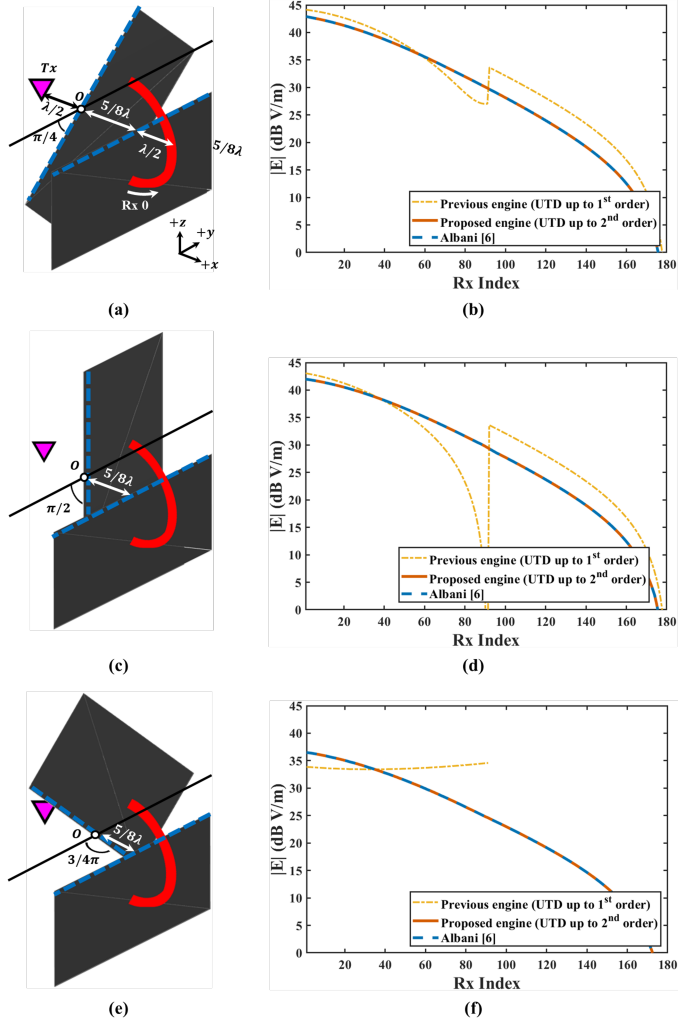


Fig. 4. Field comparison for the three Albani validation geometries with edge angles of $\psi = \pi/4$, $\pi/2$, and $3\pi/4$. The curves compare (i) our previous RT engine, which includes UTD up to first order, (ii) the proposed engine, which includes second-order UTD for MD, and (iii) Albani's reference solution [7]. The prior engine exhibits discontinuities near transition regions, whereas the proposed engine closely matches Albani's model once MD is included.

10^4 double-diffraction candidate configurations from feasible wedge-pair geometries in arbitrary 3-D configurations. On this diagnostic batch, the C–A baseline solver exhibited the expected behavior for regular configurations: the accepted regular solutions satisfied the Keller condition in the post-convergence physical validity check, and the iteration count remained low with 5/50/95 percentiles of 0/3/6. These results confirm that the C–A solver is generally fast and reliable for ordinary differentiable MD paths in practical urban geometries.

However, the same diagnostic batch also contained 40 singularity-sensitive candidate paths that approached the $Q_1 \approx Q_2$ collapse region and terminated by the step-norm criterion before satisfying the Keller-stationary point. This observation motivates the SGN stabilization developed below, which preserves the fast C–A convergence behavior for regular candidates while selectively stabilizing singularity-sensitive paths.

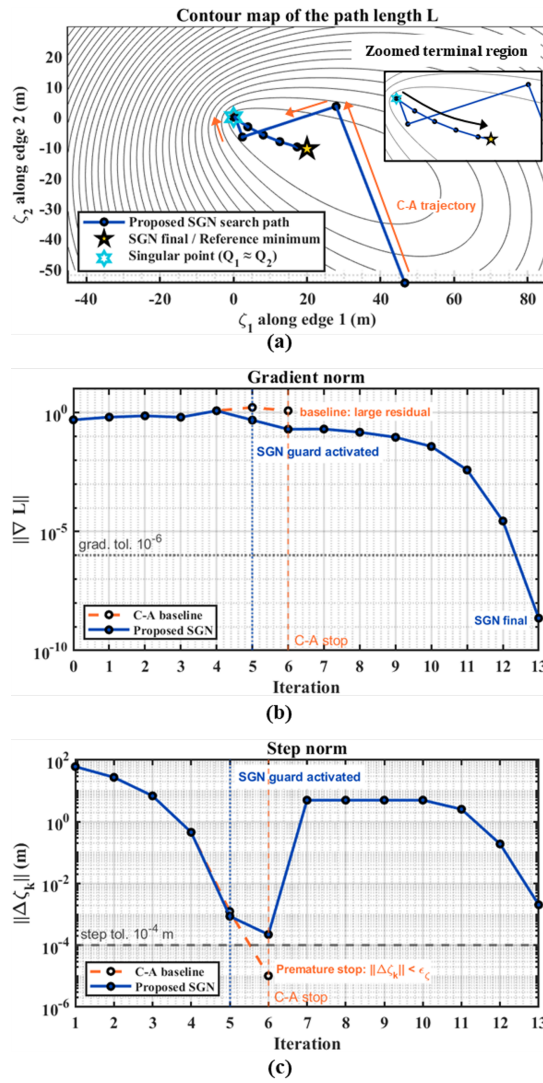


Fig. 5. Singularity-induced premature termination of the C-A baseline and recovery by the proposed SGN optimizer for a representative double-diffraction diagnostic geometry. The contour map in (a) is plotted in the (ζ_1, ζ_2) space with $Q_i(\zeta_i) = O_i + \zeta_i \hat{\zeta}_i$. The diagnostic geometry is $P' = (39.328, 92.657, 15)$ m, $P = (-106.375, 190.737, 2)$ m, $O_1 = O_2 = (0, 0, 18)$ m, $\hat{\zeta}_1 = (0.016, 0.999, 0)$, and $\hat{\zeta}_2 = (0.999, -0.014, 0)$. The cyan marker denotes the $Q_1 \approx Q_2$ collapse region where the C-A baseline prematurely terminates, whereas the yellow star denotes the Keller-stationary reference minimum. The step- and gradient-norm histories in (b) and (c) show that the C-A baseline satisfies the step-norm stopping criterion $\|\Delta\zeta^k\| < 10^{-4}$ m near the collapse region while retaining a large raw-gradient residual. The proposed SGN detects the collapse-risk trial step, moves away from the collapse region, and converges to the Keller-stationary point.

B. Singularity-Induced Premature Termination

Fig. 5 shows a representative diagnostic case whose geometry is specified in the caption. The case was selected because the two wedge lines intersect at $O_1 = O_2$, so the Newton iteration can drive the two diffraction points Q_1 and Q_2 toward the common intersection point. As $\|Q_2 - Q_1\|$ becomes nearly zero, the path-length objective becomes non-differentiable and the Hessian becomes ill-conditioned or undefined, as also noted in [17]. This makes the geometry singularity-sensitive and allows us to examine how the C-A baseline can be attracted toward the $Q_1 = Q_2$ collapse point instead of the

finite-separation reference minimum.

As highlighted by the orange C-A baseline trajectory in Fig. 5(a), the C-A trajectory approaches the cyan $Q_1 \approx Q_2$ collapse point and terminates in its neighborhood. At iteration 5, the separation between the two diffraction points is already 1.01482×10^{-5} m, indicating that the candidate solution is nearly collapsed. The Armijo line search then accepts only a very small cost-decreasing update near the cusp, so the next update has a step norm of $\|\Delta\zeta^6\| = 1.02502 \times 10^{-5}$ m $< \epsilon_\zeta = 10^{-4}$ m, causing the C-A baseline solver to terminate at iteration 6 by the step-norm criterion, as shown in Fig. 5(b).

However, the terminal point is not a valid solution: the raw-gradient norm remains large at 1.20386, as shown in Fig. 5(c), and the Keller-law condition is also not satisfied. For the same geometry, the finite-separation Keller-stationary reference point is $\zeta_{\text{ref}}^* = [19.95, -10.35]^T$ m, marked by the yellow star in Fig. 5(a). Thus, the C-A endpoint near the cyan $Q_1 \approx Q_2$ collapse point is a premature numerical termination rather than the desired Keller-stationary solution. This motivates the SGN stabilization described in the next subsection.

C. Singularity-Guarded Newton Stabilization

The goal of SGN is to preserve the fast convergence behavior of the C-A baseline solver for ordinary differentiable paths, while activating additional stabilization only when the raw line-search step approaches the $Q_1 \approx Q_2$ collapse region. This selective activation keeps regular paths on the original C-A update path and confines the additional computation to guard-active paths.

The proposed SGN stabilization is organized into the following three-step layered safeguard around the $Q_1 \approx Q_2$ singular region:

- (i) **Segment-wise singularity guard:** This guard detects whether the candidate endpoint or the line-search segment approaches the non-differentiable $Q_1 = Q_2$ collapse set. When the guard is activated, SGN restores the lowest-cost previously accepted iterate whose middle-segment distance remains above the guard threshold, i.e., $\|Q_2 - Q_1\| > \delta$, so that the subsequent stabilization starts from a non-collapsed configuration.
- (ii) **Locally smoothed path-length objective:** The raw distance term $\|x\|$ is replaced by $\sqrt{\|x\|^2 + \epsilon_s^2}$ near the collapse region to alleviate the derivative singularity associated with the zero-length middle segment [33].
- (iii) **Regularized trust-region Newton update:** A stabilized search direction is computed using $H_{\epsilon_s} + \mu I$, and the trust-region radius bounds the step size to prevent a large collapse-oriented Newton update [34], [35].

After the iterate moves back to a safe finite-separation region, SGN switches back to the original unsmoothed C-A objective for final raw polishing and physical validation. The first component, the segment-wise singularity guard, is detailed below. Although the Armijo backtracking line search ensures a cost-decreasing trial endpoint, it does not check whether the finite line-search segment itself passes through, or too close to, the non-differentiable $Q_1 = Q_2$ collapse set.

Therefore, SGN monitors the minimum separation between the two diffraction points along the actual trial segment and activates the singularity guard when this distance falls below a prescribed threshold. For the current iterate ζ^k , search direction $\mathbf{p} = [p_1, p_2]^T$, and trial step size α , the intermediate vector between the two diffraction points along the trial segment is

$$\mathbf{r}(\tau) = Q_2(\zeta_2^k + \tau p_2) - Q_1(\zeta_1^k + \tau p_1) = \mathbf{r}_0 + \tau \mathbf{v}, \quad 0 \leq \tau \leq \alpha, \quad (12)$$

where

$$\mathbf{r}_0 = Q_2^k - Q_1^k, \quad \mathbf{v} = \hat{\zeta}_2 p_2 - \hat{\zeta}_1 p_1. \quad (13)$$

Because $\mathbf{r}(\tau)$ is affine in τ , the closest point on the finite line-search segment can be computed analytically as

$$\tau^* = \min \left\{ \alpha, \max \left\{ 0, -\frac{\mathbf{r}_0^T \mathbf{v}}{\|\mathbf{v}\|^2} \right\} \right\}. \quad (14)$$

The minimum middle-segment distance along the actual trial step is then

$$d_{\min} = \|\mathbf{r}_0 + \tau^* \mathbf{v}\|. \quad (15)$$

The singularity guard is activated when

$$d_{\min} \leq \delta, \quad (16)$$

where δ is the prescribed guard length. In this case, the trial step is regarded as approaching the $Q_1 \approx Q_2$ collapse region and is not accepted as a regular C–A update.

When the singularity guard is activated, SGN restores the lowest-cost previously accepted iterate that maintains sufficient finite separation between the two diffraction points. This prevents the subsequent local update from starting inside the collapse region. SGN then uses a locally smoothed path-length objective by replacing each Euclidean distance term with

$$\|\mathbf{x}\|_{\epsilon_s} = \sqrt{\|\mathbf{x}\|^2 + \epsilon_s^2}. \quad (17)$$

The corresponding smoothed objective is

$$L_{\epsilon_s}(\zeta) = \|Q_1 - P'\|_{\epsilon_s} + \|Q_2 - Q_1\|_{\epsilon_s} + \|P - Q_2\|_{\epsilon_s}. \quad (18)$$

This smoothed objective remains close to the original path-length objective while replacing the undefined derivative at zero separation with a bounded derivative near the collapse region.

A regularized trust-region Newton update is then used to compute a stable search direction:

$$(\mathbf{H}_{\epsilon_s} + \mu \mathbf{I}) \mathbf{p} = -\mathbf{g}_{\epsilon_s}, \quad \|\mathbf{p}\| \leq \Delta, \quad (19)$$

where $\mathbf{g}_{\epsilon_s} = \nabla L_{\epsilon_s}$, $\mathbf{H}_{\epsilon_s} = \nabla^2 L_{\epsilon_s}$, μ is the regularization parameter, and Δ is the trust-region radius. The regularization term $\mu \mathbf{I}$ stabilizes the Hessian model when the local curvature becomes unreliable near the collapse region, and the trust-region constraint prevents a large collapse-oriented Newton step [34]–[36]. The damping parameter is increased when the regularized system is singular, indefinite, ill-conditioned, non-finite, or does not produce a descent direction.

The stabilized step is accepted only when it remains outside the singularity guard and decreases the smoothed objective.

The final path is accepted only when the step norm, raw-gradient norm, and Keller-law residual are all below tolerances on the order of 10^{-4} , preventing premature acceptance based solely on a small step near the singular collapse region.

D. Path-Recovery Results and Physical Validation

Fig. 5 also shows the SGN result for the representative singularity-sensitive path. Until iteration 5, SGN follows the same raw C–A iteration as the baseline solver while the path remains in a regular region. At iteration 5, however, the next raw trial segment approaches the $Q_1 = Q_2$ collapse set, and the segment-wise singularity guard is activated because the minimum separation along the trial segment falls below the guard threshold. The unguarded C–A baseline subsequently accepts only a very small collapse-side update and stops at iteration 6 by the step-norm criterion, as indicated by the orange C–A stop marker. In contrast, SGN rejects this collapse-side raw trial, restores the best safe finite-separation iterate, and activates the smoothed trust-region stabilization. The stabilized updates then move the iterate away from the collapse region, and SGN finally converges to the finite-separation Keller-stationary solution at iteration 13.

The stabilized updates move the iterate away from the collapse region and guide it toward the finite-separation minimum of the raw path-length objective. For the representative path, the final SGN solution is located at

$$(\zeta_1, \zeta_2) = (19.95, -10.35), \quad (20)$$

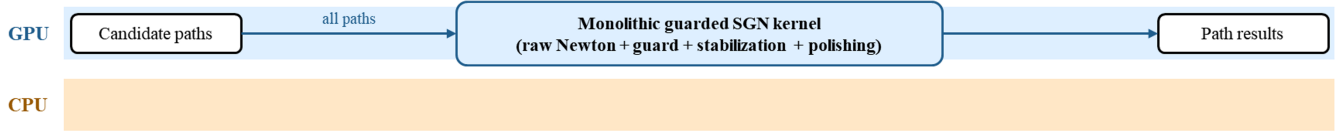
which coincides with the reference minimum shown in the contour map. The final raw-gradient norm is reduced to 2.28589×10^{-9} , and the Keller condition is satisfied ($\beta'_1 = \beta_1 = 27.3949^\circ$ and $\beta'_2 = \beta_2 = 62.5257^\circ$). This confirms that SGN recovers the valid finite-separation Keller-stationary path that the unguarded C–A baseline solver misses in this singularity-sensitive case.

It should be noted that the selective design of SGN preserves the efficiency of the C–A baseline solver for regular paths. For the 47,505 paths already solved by the C–A baseline, SGN kept the same 5/50/95 iteration percentiles as the baseline solver, namely 0/3/6. The additional computational cost was concentrated on the newly recovered 40 singularity-sensitive paths, for which the 5/50/95 iteration percentiles were 9.2/13/18.8. Therefore, SGN improves convergence robustness by preventing premature termination near the collapse region, while preserving the fast convergence behavior of the C–A baseline for ordinary double-diffraction paths.

The SGN stabilization described in this section introduces conditional branching and additional stabilized updates only for guard-active paths. Although this selective design is efficient at the algorithmic level, a direct GPU implementation can still suffer from warp divergence and dispatch overhead. The next section first reorganizes the SGN execution flow for GPU-oriented acceleration and then integrates the resulting optimizer into the full 3-D image-theory ray-tracing pipeline.

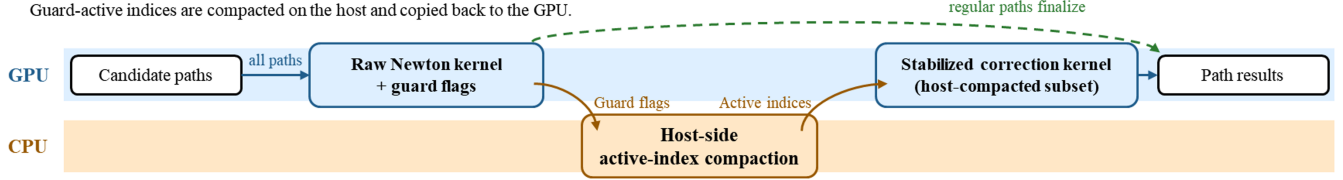
(a) Monolithic guarded SGN kernel

All paths execute the full guarded SGN logic in a single GPU kernel.



(b) Host-compacted split SGN

Guard-active indices are compacted on the host and copied back to the GPU.



(c) Proposed device-compacted split SGN

Active indices remain compacted on the device; the host receives only the active count.

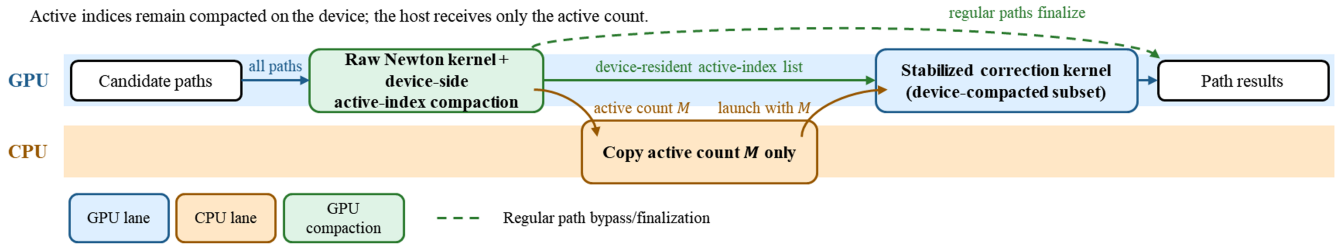


Fig. 6. Domain-separated execution pipelines for the three GPU implementations of the singularity-guarded Newton path optimizer. (a) The monolithic guarded SGN executes the full guarded optimization logic for all candidate paths in a single GPU kernel. (b) The host-compacted split SGN separates regular and guard-active paths, performs active-index compaction on the host, and launches the stabilized correction kernel for the host-compacted subset. (c) The proposed device-compacted split SGN performs active-index compaction directly on the GPU, keeps the compacted index list in device memory, and launches the stabilized correction kernel for the device-compacted subset.

IV. ACCELERATION AND INTEGRATION OF MULTI-DIFFRACTION IN A FULL 3-D IT-RT FRAMEWORK

In this section, we integrate the multi-diffraction algorithm and the SGN optimizer developed in the previous sections into our accelerated IT-RT engine. Compared with the mmWave scenarios targeted by our earlier engines [28], diffraction has a more pronounced impact in the 6G upper-mid band, making it necessary to extend the RT engine from first-order diffraction to MD support for 6G coverage and system analysis [2], [3].

We first optimize the GPU execution structure of the robust SGN path optimizer, and then describe the full 3-D multi-diffraction pipeline, including VT construction, path tracing (PT), shadowing test (ST), compact valid-path dataflow, and end-to-end runtime and memory benchmarks.

A. GPU-Oriented Acceleration of the Robust Path Optimizer

Fig. 6 illustrates the execution-flow refactoring used to reduce the GPU overhead of the singularity-guarded Newton optimizer. The proposed SGN optimizer improves numerical robustness near singular double-diffraction configurations, but unlike the C–A baseline solver, it introduces conditional branching associated with the singularity guard. When the branch activation differs across candidate paths, threads within the same warp may follow different execution paths, causing warp divergence and additional branch overhead, which can reduce the effective GPU throughput. To mitigate this issue,

we use the throughput of the GPU-implemented C–A baseline solver as the reference performance and progressively optimize the execution structure of the proposed SGN optimizer in three stages: monolithic guarded SGN, host-compacted split SGN, and proposed device-compacted split SGN.

The first stage, monolithic guarded SGN, executes the entire SGN procedure for every candidate path within a single GPU kernel. Each thread is assigned to one candidate path and sequentially performs the raw Newton iteration, singularity-guard evaluation, and stabilization step inside the same monolithic kernel. This implementation applies the same guarded optimization logic to all candidate paths and therefore serves as a reference implementation for verifying the numerical robustness of SGN. However, most candidate paths are regular paths that do not activate the singularity guard. Consequently, applying the full guarded logic to all paths introduces unnecessary guard evaluations and branch decisions, and when only a subset of threads within the same warp enters the stabilization branch, warp divergence occurs. Therefore, although the monolithic guarded SGN preserves robustness, it suffers from a significant throughput loss compared with the C–A baseline solver.

The second stage, host-compacted split SGN, separates regular paths from guard-active paths to reduce the unnecessary guarded execution introduced by the monolithic structure. Here, a guard-active path denotes a candidate path for which

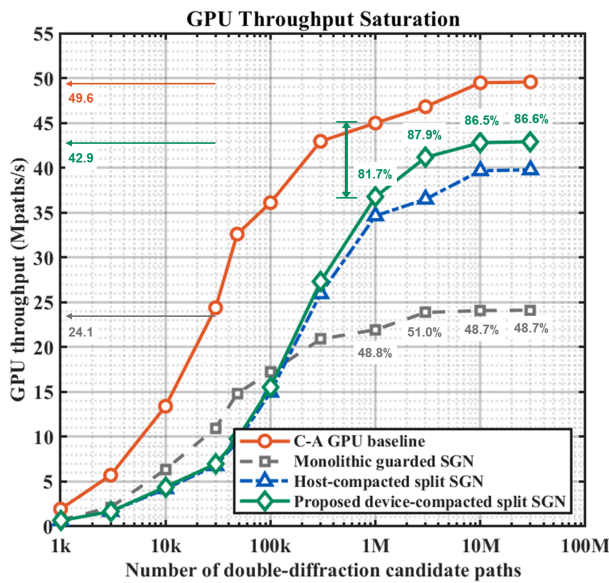


Fig. 7. GPU throughput saturation for double-diffraction path optimization as the number of candidate paths increases. The Carluccio–Albani GPU baseline does not include singularity-guarded branching and is used as the throughput reference. The three guarded implementations progressively reduce the overhead of the singularity-aware stabilization: monolithic guarded execution, host-compacted split execution, and the proposed device-compacted split execution. The percentage annotations indicate the throughput retained relative to the Carluccio–Albani GPU baseline in the high-throughput regime.

the singularity guard is activated during the raw Newton phase and additional stabilization is required. First, the raw Newton kernel is applied to all candidate paths. Regular paths that do not activate the singularity guard are finalized immediately, whereas guard-active paths are marked using guard flags. The guard-flag array is then copied from the device to the host. On the host, only the indices of guard-active paths are selected and compacted into a contiguous active-index list through host-side active-index compaction. The generated active-index list is then copied back to the device, and the stabilized correction kernel is launched only for this host-compacted subset.

In this structure, the stabilized correction kernel performs stabilization updates only for candidate paths classified as guard-active in the raw Newton phase. Therefore, the host-compacted split SGN limits the correction computation to the guard-active subset and reduces unnecessary work compared with the monolithic implementation, which applies the full guarded logic to all paths. However, before launching the correction kernel, this approach requires the device-to-host (D2H) transfer of the guard-flag array, host-side active-index filtering and compaction, the host-to-device (H2D) transfer of the compacted index list, and synchronization. Thus, although the amount of stabilized correction computation is reduced, dispatch overhead due to host-side compaction still remains.

The third stage, proposed device-compacted split SGN, reduces this host-side active-index compaction overhead by compacting the guard-active path indices directly on the GPU. When a guard-active path is detected in the raw Newton kernel, its path index is appended directly to a device-resident active-index list. This active-index list remains in GPU memory, and only the number of guard-active paths, denoted by M , is

transferred to the host. The stabilized correction kernel is then launched for the device-compacted subset using the active-index list that is already stored in device memory, rather than an index list copied back from the host. Consequently, regular paths are finalized through the fast raw-Newton execution path, while the additional stabilized correction cost scales with the number of guard-active paths rather than with the total number of candidate paths. Moreover, the proposed structure alleviates the host-side filtering bottleneck by eliminating the full guard-flag transfer and CPU-side index compaction.

Fig. 7 shows how closely the three stages of GPU execution-structure optimization approach the throughput of the C–A baseline solver. In the low path-count regime, all solvers exhibit low throughput because kernel launch, device-host synchronization, and dispatch overhead account for a relatively large fraction of the total execution time. However, in the large-scale complex urban scenarios targeted in this work, the number of MD candidate paths is generally on the order of several hundred thousand or more. Accordingly, the following comparison focuses on how much GPU performance each SGN implementation recovers relative to the C–A GPU baseline in the throughput-saturation regime, which is the practical region of interest for the targeted large path-count simulations.

The C–A GPU baseline, which does not include singularity-guarded branching, achieves the highest throughput and saturates at approximately 49.6 Mpaths/s in the large path-count regime. In contrast, the monolithic guarded SGN saturates at approximately 24.1 Mpaths/s because it applies the full guarded execution to every candidate path. This corresponds to approximately 48.7–51.0% of the C–A baseline throughput. Thus, although the monolithic guarded SGN preserves the numerical robustness of SGN near singular configurations, it maintains only about half of the baseline throughput due to branch overhead and warp divergence.

The host-compacted split SGN achieves higher throughput than the monolithic guarded SGN by separating regular paths from guard-active paths. This improvement occurs because most regular paths are finalized directly in the raw Newton phase without passing through the stabilized correction kernel, while the correction computation is restricted to the guard-active subset. However, because active-index compaction is still performed on the host, a throughput gap from the C–A GPU baseline remains, indicating that active-index compaction should be moved inside the device to further approach baseline throughput.

The proposed device-compacted split SGN performs guard-active index compaction directly on the GPU and reduces the host-side filtering overhead. As a result, it achieves approximately 42.9 Mpaths/s in the large path-count regime, corresponding to approximately 81.7–87.9% of the C–A baseline throughput. In particular, in the saturation region, it retains approximately 86.5–86.6% of the baseline throughput, which represents a substantial performance recovery compared with the 48.7–51.0% retained by the monolithic guarded SGN. Therefore, the proposed device-compacted split SGN preserves the singularity-aware robustness of the SGN optimizer while recovering throughput close to that of the C–A baseline solver through GPU-oriented execution-structure optimization.

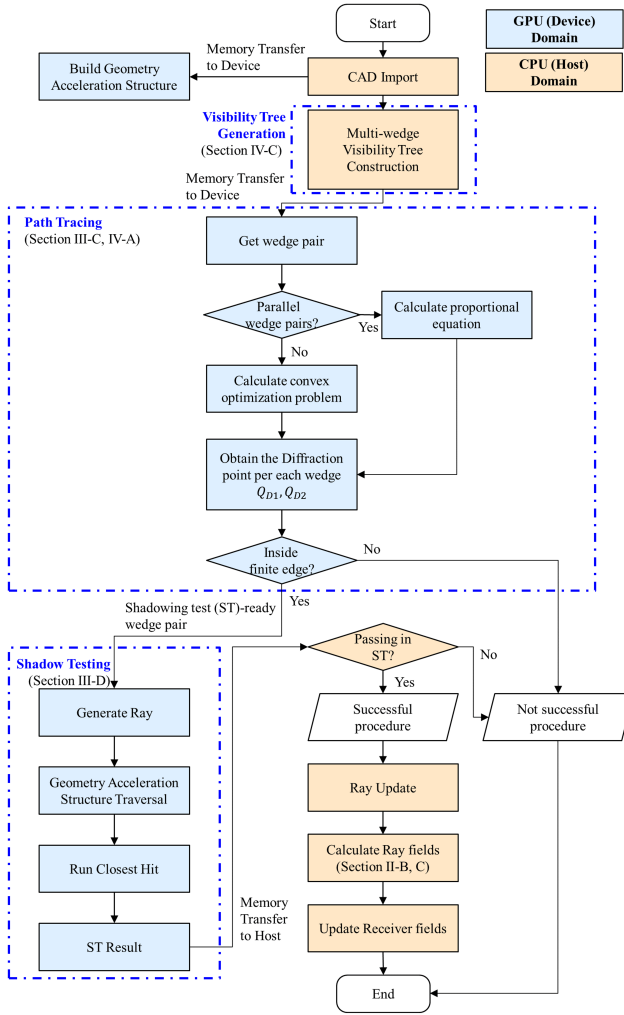


Fig. 8. Flowchart showing the procedure of the proposed multi-diffraction RT method, as well as H2D memory transfers.

B. Full 3-D Multi-Diffraction Pipeline and Host-Device Data Layout

Fig. 8 summarizes the full 3-D multi-diffraction RT pipeline and highlights the CPU/GPU acceleration regions within the IT-RT engine. As demonstrated in [2], the geometric structures to be imported are prepared as triangular facets in Blender [37], where building shapes and heights are obtained from OpenStreetMap [38] and the Korean terrain is provided by V-World [39]. Consequently, the assembled geometry is then loaded into the RT engine and transferred to the GPU, where NVIDIA OptiX builds the geometry acceleration structure (GAS) used for PT and ST.

Subsequently, for each feasible wedge pair, we first allocate a memory buffer as part of the VT construction. We then compute the diffraction points Q_1 and Q_2 on the first and second wedges via PT, and finalize the MD ray by performing the ST to check whether any geometry occludes the $Tx-Q_1-Q_2-Rx$ path. The following subsections describe these components in execution order, starting from VT construction and then proceeding to path tracing, shadowing test, and compact valid-path field calculation.

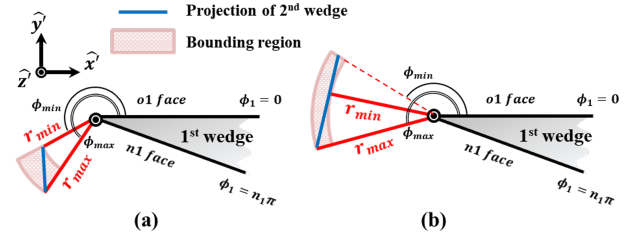


Fig. 9. Bounding region of anxel beam for diffraction-diffraction case.

Host-Device Data Layout. During the GAS construction, the assembled geometry is uploaded to GPU global memory as contiguous array-of-structures (AoS) buffers. These buffers store the RT primitives, including vertex coordinates, Rx point coordinates, face normal vectors, and triangle-index triplets, together with wedge data consisting of the wedge endpoints and the associated o - and n -face indices.

MD ray candidates are also stored as a contiguous AoS buffer. Each candidate entry contains the PT-computed diffraction points Q_1 and Q_2 , the VT-determined wedge-pair indices, and the receiver index associated with the path. All buffers are passed as raw device pointers through the kernel launch parameters, enabling direct dereferencing in the PT/ST kernels.

Benchmarking Environment. We used an NVIDIA RTX A6000 (Ampere, 48 GB), CUDA 11.8, and NVIDIA OptiX 7.2.0 [40] on Windows Server 2019 Standard (Version 1809). The host system is a dual Intel Xeon E5-2687v4 (3.00 GHz) with 512 GB RAM. The CPU baseline employs OpenMP with 16 threads. All in-house codes were implemented in C++ with the NVIDIA OptiX framework.

C. Multi-Edge Visibility Tree Construction Using DABS

In IT-RT, the VT represents the visibility relationships among the Tx, facets, edges, and Rx. Its size increases with scenario complexity, i.e., with the number of meshes and wedges in arbitrary 3-D geometries, and further grows with higher bounce orders. Therefore, efficient VT construction is essential for large urban simulations. An anxel-beam denotes the bounded angular or angular-edge-parameter beam region used in AZB-type IT-RT to enclose candidate facets or edges from a source or image source. Although our previous work [41] efficiently defined such anxel-beam bounds for mmWave-band RT, MD modeling was beyond its scope.

In the diffraction-diffraction case, we adopt a ϕ - t bounding region parameterized by $(\phi_{\min}, \phi_{\max}, t_{\min}, t_{\max})$ following the authors' geometry and notation for the arc of the diffraction bounding region [41, Figs. 6-8], thereby defining the DABS.

In this ϕ - t system, $\phi_{\min}$ and $\phi_{\max}$ can be obtained from the coordinates of the two endpoints of the orthographic projection of the second wedge, as illustrated in Fig. 9. By contrast, computing $t_{\min}$ and $t_{\max}$ in a deterministic manner is challenging because, as discussed in Section II, no closed-form solution exists for MD paths. Instead, we compute lower and upper bounds on t , from which $t_{\min}$ and $t_{\max}$ can be efficiently determined.

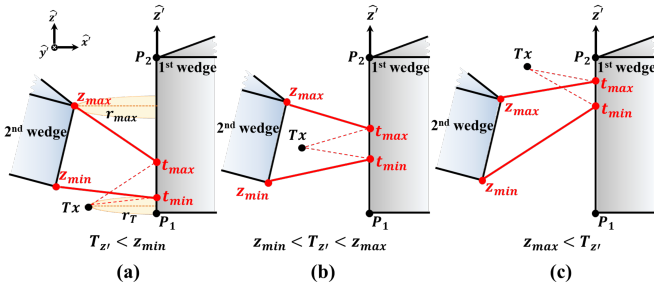


Fig. 10. Three cases for the relative position for diffraction-diffraction case.

For diffraction-diffraction, Fig. 10 illustrates the relative positions of the Tx and the second wedge's bounding region. Three cases are considered: (1) $T_{z'} < z_{min}$; (2) $z_{min} < T_{z'} < z_{max}$; and (3) $z_{max} < T_{z'}$. On the second wedge, the parameter t is confined to values that satisfy Keller's law within its z' -interval $P_{2z'}$. For a point R on the second wedge, t is given by

$$t = \frac{T_{z'} - P_{1z'}}{P_{2z'} - P_{1z'}} + \frac{r_T}{P_{2z'} - P_{1z'}} \frac{R_{z'} - T_{z'}}{r_R + r_T} \quad (21)$$

where $T_{z'}$ and r_T denote the Tx z' -position and its radial distance from the z' -axis; $R_{z'}$ and r_R are the corresponding quantities for the point R ; and $P_{1z'}$, $P_{2z'}$ are the lower and upper z' -endpoints of the first wedge, respectively. Note that the only unknowns are $R_{z'}$ and r_R .

From (21), t increases monotonically with larger $R_{z'}$ and smaller r_R (and decreases with smaller $R_{z'}$ and larger r_R). Therefore, for each configuration in Fig. 10 (Cases 1–3), the bounds t_{min} and t_{max} are obtained by bounding the second term of (21), $(R_{z'} - T_{z'})/(r_R + r_T)$, over $R_{z'} \in [z_{min}, z_{max}]$ and $r_R \in [r_{min}, r_{max}]$ according to the sign of $R_{z'} - T_{z'}$.

Case 1 ($T_{z'} < z_{min}$):

Here $R_{z'} - T_{z'} > 0$, so the second term is positive. We bound it as $\left(\frac{R_{z'} - T_{z'}}{r_R + r_T}\right)_{min} \geq \frac{z_{min} - T_{z'}}{r_{max} + r_T} > 0$, and $\left(\frac{R_{z'} - T_{z'}}{r_R + r_T}\right)_{max} \leq \frac{z_{max} - T_{z'}}{r_{min} + r_T}$. Substituting into (21) gives

$$t_{min} = \frac{T_{z'} - P_{1z'} + \frac{r_T(z_{min} - T_{z'})}{r_{max} + r_T}}{P_{2z'} - P_{1z'}}, \quad (22a)$$

$$t_{max} = \frac{T_{z'} - P_{1z'} + \frac{r_T(z_{max} - T_{z'})}{r_{min} + r_T}}{P_{2z'} - P_{1z'}}. \quad (22b)$$

Case 2 ($z_{min} < T_{z'} < z_{max}$):

The second term changes sign across the interval. We bound $0 > \left(\frac{R_{z'} - T_{z'}}{r_R + r_T}\right)_{min} \geq \frac{z_{min} - T_{z'}}{r_{max} + r_T}$, and $0 < \left(\frac{R_{z'} - T_{z'}}{r_R + r_T}\right)_{max} \leq \frac{z_{max} - T_{z'}}{r_{min} + r_T}$. These bounds yield

$$t_{min} = \frac{T_{z'} - P_{1z'} + \frac{r_T(z_{min} - T_{z'})}{r_{min} + r_T}}{P_{2z'} - P_{1z'}}, \quad (22c)$$

$$t_{max} = \frac{T_{z'} - P_{1z'} + \frac{r_T(z_{max} - T_{z'})}{r_{min} + r_T}}{P_{2z'} - P_{1z'}}. \quad (22d)$$

Algorithm 1 Determination of t_{min} and t_{max} with o-/n-face intersection for diffraction-diffraction

Input Common tuple $\Pi = (T_{z'}, r_T, P_{1z'}, P_{2z'})$

Input Face tuples $\mathbf{b}^o = (z_{min}^o, z_{max}^o, r_{min}^o, r_{max}^o)$, $\mathbf{b}^n = (z_{min}^n, z_{max}^n, r_{min}^n, r_{max}^n)$

Output Final interval $[t_{min}^*, t_{max}^*]$

```

1: function TBOUNDS( $\Pi, \mathbf{b}$ )
2:    $(T_{z'}, r_T, P_{1z'}, P_{2z'}) \leftarrow \Pi$ 
3:    $(z_{min}, z_{max}, r_{min}, r_{max}) \leftarrow \mathbf{b}$ 
4:   if  $T_{z'} < z_{min}$  then ▷ Case 1
5:      $t_{min} \leftarrow \frac{T_{z'} - P_{1z'} + \frac{r_T(z_{min} - T_{z'})}{r_{max} + r_T}}{P_{2z'} - P_{1z'}}$ 
6:      $t_{max} \leftarrow \frac{T_{z'} - P_{1z'} + \frac{r_T(z_{max} - T_{z'})}{r_{min} + r_T}}{P_{2z'} - P_{1z'}}$ 
7:   else if  $z_{min} < T_{z'} < z_{max}$  then ▷ Case 2
8:      $t_{min} \leftarrow \frac{T_{z'} - P_{1z'} + \frac{r_T(z_{min} - T_{z'})}{r_{min} + r_T}}{P_{2z'} - P_{1z'}}$ 
9:      $t_{max} \leftarrow \frac{T_{z'} - P_{1z'} + \frac{r_T(z_{max} - T_{z'})}{r_{min} + r_T}}{P_{2z'} - P_{1z'}}$ 
10:  else ▷ Case 3
11:     $t_{min} \leftarrow \frac{T_{z'} - P_{1z'} + \frac{r_T(z_{min} - T_{z'})}{r_{min} + r_T}}{P_{2z'} - P_{1z'}}$ 
12:     $t_{max} \leftarrow \frac{T_{z'} - P_{1z'} + \frac{r_T(z_{max} - T_{z'})}{r_{max} + r_T}}{P_{2z'} - P_{1z'}}$ 
13:  end if
14:  return  $t_{min}, t_{max}$ 
15: end function
16:  $(t_{min,o}, t_{max,o}) \leftarrow \text{TBOUNDS}(\Pi, \mathbf{b}^o)$ 
17:  $(t_{min,n}, t_{max,n}) \leftarrow \text{TBOUNDS}(\Pi, \mathbf{b}^n)$ 
18:  $t_{min}^* \leftarrow \max(t_{min,o}, t_{min,n})$ 
19:  $t_{max}^* \leftarrow \min(t_{max,o}, t_{max,n})$ 
20: return  $t_{min}^*, t_{max}^*$ 

```

Case 3 ($z_{max} < T_{z'}$):

Now $R_{z'} - T_{z'} < 0$, so the second term is negative throughout. We bound $0 > \left(\frac{R_{z'} - T_{z'}}{r_R + r_T}\right)_{min} \geq \frac{z_{min} - T_{z'}}{r_{max} + r_T}$, and $0 > \frac{z_{max} - T_{z'}}{r_{min} + r_T} \geq \left(\frac{R_{z'} - T_{z'}}{r_R + r_T}\right)_{max}$. Consequently,

$$t_{min} = \frac{T_{z'} - P_{1z'} + \frac{r_T(z_{min} - T_{z'})}{r_{min} + r_T}}{P_{2z'} - P_{1z'}}, \quad (22e)$$

$$t_{max} = \frac{T_{z'} - P_{1z'} + \frac{r_T(z_{max} - T_{z'})}{r_{max} + r_T}}{P_{2z'} - P_{1z'}}. \quad (22f)$$

From (22), we compute the bounds of t for the second wedge's o-face and n-face, and then tighten the interval via $\max(t_{min,o}, t_{min,n})$ and $\min(t_{max,o}, t_{max,n})$, which yields the final admissible range of t for the diffraction-diffraction bounding region. Algorithm 1 consolidates the above discussion.

Finally, to accelerate the PT/ST block, we construct a bounding volume hierarchy (BVH) with NVIDIA OptiX and

Algorithm 2 Multi-Diffraction Path Tracing

Input Tx, Rx, wedge endpoints A_1, B_1 for W_1 and A_2, B_2 for W_2
Output Q_1, Q_2 (diffraction points on W_1, W_2)

- 1: $u_1 \leftarrow \text{normalize}(B_1 - A_1)$
- 2: $u_2 \leftarrow \text{normalize}(B_2 - A_2)$
- 3: **if** ISPARALLEL(u_1, u_2) **then** ▷ Closed Form
- 4: $u \leftarrow u_1$
- 5: $p_1 \leftarrow \text{proj}_{(A_1, u)}(\text{Tx}), \quad p_2 \leftarrow \text{proj}_{(A_2, u)}(\text{Tx})$
- 6: $p_3 \leftarrow \text{Rx} + u(u^\top(\text{Tx} - \text{Rx}))$
- 7: $d_1 \leftarrow \|\text{Tx} - p_1\|, \quad d_2 \leftarrow \|p_2 - p_1\|, \quad d_3 \leftarrow \|p_3 - p_2\|$
- 8: $\alpha \leftarrow \frac{u^\top(\text{Rx} - p_3)}{d_3}$
- 9: $\lambda \leftarrow \frac{d_1 + d_2 + d_3}{d_1 + d_2 + d_3}$
- 10: $\Delta_1 \leftarrow \lambda d_1, \quad \Delta_2 \leftarrow \alpha - \lambda d_3$
- 11: $Q_1 \leftarrow p_1 + u \Delta_1, \quad Q_2 \leftarrow p_2 + u \Delta_2$
- 12: **else**
- 13: $(Q_1, Q_2) \leftarrow \text{CONVEXOPTIMIZATION}(\text{Tx}, \text{Rx}, W_1, W_2)$
▷ SGN, Section III-C and IV-A
- 14: **end if**
- 15: $\text{hit} \leftarrow \text{INSIDEEDGE}(W_1, W_2, Q_1, Q_2)$
- 16: **if** hit **then**
- 17: **return** Q_1, Q_2
- 18: **else**
- 19: **return** INVALID, INVALID
- 20: **end if**

use it as the acceleration structure to execute the PT/ST block described in Section IV-D. Building on this, the set of Rx's residing inside each DABS is attached to—and detached from—the constructed VT during traversal, so that the PT/ST block can efficiently discover MD rays.

The advantages of DABS for accelerating computation are as follows. (1) In arbitrary 3-D wedges where MD paths cannot be determined in a closed-form manner, DABS removes unnecessary edges during VT construction without complex computations, thereby enabling full 3-D IT-RT at large scale. (2) Because each subtree (wedge) stemming from the VT root (Tx) is processed independently, OpenMP-based parallel execution is directly applicable. (3) DABS can recursively be used with existing methods (e.g., reflection–diffraction and diffraction–reflection [41]), enabling straightforward construction of a full 3-D IT-RT framework. Moreover, as the bounce order increases, the DABS bounding region contracts, enabling a more memory-efficient VT construction. In this study, the implementation and impact analysis of MD paths that include chained R are outside our scope and are left for future work.

D. Multi-Diffraction Path Evaluation and Shadowing Validation

The solver for the convex optimization problem in arbitrary wedge pairs has already been validated in Section II-A. However, invoking the optimizer is not necessary for all configurations. For parallel wedge pairs—such as the common vertical–vertical or horizontal–horizontal cases in urban scenes—a closed-form shortcut deterministically produces the

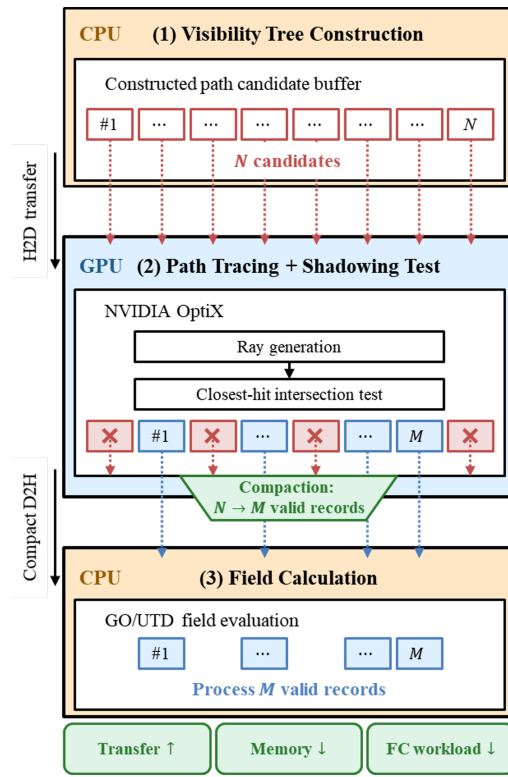


Fig. 11. Proposed compact valid-path pipeline. After VT construction, N candidate path records are transferred to the GPU for path tracing and shadowing. The shadowing test compacts the surviving records into M valid paths, where $M \ll N$, and only these compact records are transferred back to the host. Field calculation then iterates over the compact valid-path records, evaluates the UTD field contribution.

two diffraction points, substantially reducing the computational cost.

Algorithm 2 details the procedure. Briefly, for parallel wedge pairs, project Tx onto each wedge line to obtain p_1 and p_2 , align Rx with Tx along the common edge direction u to form p_3 , then distribute the 1-D distance along u to place Q_1 and Q_2 on the wedges. In all non-parallel cases, the problem is solved via the SGN-based CONVEXOPTIMIZATION routine described in Section II-A. Finally, INSIDEEDGE enforces segment validity and returns INVALID if either point lies outside the wedge span. Given the candidate points Q_1 and Q_2 , the paths $\text{Tx} - Q_1 - Q_2 - \text{Rx}$ are finalized by a ST. Our ST is implemented on the GPU using NVIDIA OptiX's closest-hit program [40], so all rays are processed in parallel; if a triangle in the mesh is hit before reaching Rx, the ST fails for that path. Near-grazing segments are handled by tracing each shadow ray only from a small offset beyond the interaction point to the end of the current propagation segment, and by checking whether the first intersected CAD element corresponds to the scheduled interaction face or edge identified during VT construction. This treatment suppresses self-intersection near bouncing points, leaving only possible floating-point round-off errors in near-parallel ray-surface configurations.

TABLE I

STAGE-WISE RUNTIME OPTIMIZATION AND FINAL MEMORY FOOTPRINT FOR THE FULL 3-D IT-RT ENGINE. PANEL (A) REPORTS THE RUNTIME REDUCTION OBTAINED BY PROGRESSIVELY APPLYING THE COMPACT DATA-TRANSFER AND COMPACT-PIPELINE OPTIMIZATIONS RELATIVE TO THE CORRESPONDING LEGACY BASELINE. PANEL (B) COMPARES THE FINAL COMPACT-PIPELINE IMPLEMENTATION FOR $R = 6, D = 1$ AND $R = 6, D = 2$, SHOWING THE ADDITIONAL OVERHEAD INTRODUCED BY DOUBLE DIFFRACTION.

(a) Stage-wise optimization relative to each legacy baseline						
Case	Implementation	Total (s)	Saved (s)	Reduction (%)	CPU WS (GB)	GPU Mem. (GB)
$R = 6, D = 1$	Legacy baseline	3725.4	0.0	0.0	18.3	20.9
	Compact D2H	3531.5	193.9	5.2	18.3	22.9
	Proposed compact pipeline	2563.0	1162.4	31.2	16.2	22.9
$R = 6, D = 2$	Legacy baseline	3794.6	0.0	0.0	26.7	20.9
	Compact D2H	3720.7	73.9	1.9	26.7	22.9
	Proposed compact pipeline	2693.4	1101.2	29.0	24.7	22.9
(b) Final compact-pipeline benchmark						
Final configuration	Total (s)	VTC (s)	PT+ST (s)	FC (s)	CPU WS (GB)	GPU Mem. (GB)
$R = 6, D = 1$	2563.0	1716.1	173.7	2.1	16.2	22.9
$R = 6, D = 2$	2693.4	1779.5	177.0	37.4	24.7	22.9
$\Delta(D = 2 - D = 1)$	+130.4 (+5.1%)	+63.4	+3.3	+35.3	+8.5	+0.0
VTC: visibility-tree construction; PT+ST: path tracing and shadowing test; FC: field calculation.						
CPU WS: peak CPU working set; GPU Mem.: peak GPU memory usage.						
Total time includes all measured runtime components.						

E. Compact Valid-Path Pipeline and End-to-End Runtime Benchmark

Fig. 11 illustrates the proposed compact valid-path pipeline, which reduces device-to-host transfer and field-calculation overhead by keeping only the shadowing-valid path records. Extending the latest version of our accelerated IT-RT framework [3], the proposed pipeline preserves the compact valid-path representation through field calculation instead of reconstructing a dense host-side buffer. As in [28], NVIDIA OptiX is used for the closest-hit shadowing test, enabling fast candidate-path validation before field calculation.

After VT construction, the CPU constructs N candidate path records and transfers them to the GPU for path tracing and shadowing. The NVIDIA OptiX closest-hit test removes blocked or invalid paths, and the surviving records are compacted into M valid paths. In dense urban environments, many candidate paths are blocked by building structures during the shadowing test, so M is typically much smaller than the number of candidate paths N , i.e., $M \ll N$. The proposed pipeline therefore transfers only the M compact valid-path records and their source indices back to the host, instead of preserving the original N -record post-shadowing format. The field-calculation stage then processes only these M valid records, evaluates the GO/UTD field contribution, and accumulates the result to the corresponding receiver. This removes unnecessary transfer, storage, and scans of invalid path records, reducing both runtime and memory overhead in large-scale urban simulations.

To evaluate both runtime and memory footprint, we used the KT Gwanak Office CAD model from our 7.5-GHz 6G upper-mid-band urban campaign [2], [3], as shown in [2, Fig. 5]. The model contains 5,617 facets and 4,031 wedges over an approximately $750 \text{ m} \times 400 \text{ m}$ area. For the simulation, the Tx is placed at Tx2 in [2, Fig. 5], and 10,000 Rx's are uniformly distributed over the simulation area at a height of 2 m. This CAD scenario has been measurement-validated in our previous 7.5-GHz upper-mid-band studies, where RT predictions were

compared with urban measurement data in terms of path loss and delay spread, and sensitivity analyses on key modeling parameters were performed, confirming the similarity between the simulated and measured propagation characteristics [2], [3]. With the maximum bounce order set to six, we evaluate two configurations, $R = 6, D = 1$ and $R = 6, D = 2$. This scenario was chosen as a stringent urban-scale benchmark for the proposed MD-integrated IT-RT engine. In our comparison, the commercial IT-based ray tracer WinProp, which relies on OpenMP-based CPU parallelization, required more than seven days for a smaller CAD scenario and did not complete the present setup within a practical runtime [3]. This reflects the high computational cost of image-theory-based RT for large-area many-receiver simulations, which has often limited its practical use to point-to-point or small-scale analyses [24]. Thus, completing this $R = 6, D = 2$ simulation provides a meaningful demonstration of the proposed engine's scalability for full 3-D multi-diffraction over arbitrary urban geometries.

Table I summarizes the stage-wise runtime optimization and memory footprint of the developed MD-integrated accelerated IT-RT engine, which explicitly computes full 3-D multi-diffraction paths over arbitrary wedge-pair configurations in large urban CAD geometries, without reducing the problem to 2-D or parallel-wedge cases. Compared with the previous full-buffer implementation, the proposed compact valid-path pipeline in Fig. 11 is the final implementation that applies both compact D2H transfer and compact field calculation, while the compact D2H preset is evaluated as an intermediate stage that applies only compact valid-path copying and does not yet propagate the compact representation to field calculation.

Because the compact D2H preset mainly reduces the GPU-to-host memory-copy payload while the subsequent field-calculation stage still consumes the original full-buffer format, its end-to-end runtime improvement is moderate. As shown in Table I(a), compact D2H reduces the total runtime by 193.9 s for $R = 6, D = 1$ and by 73.9 s for $R = 6, D = 2$, corresponding to reductions of 5.2% and 1.9%, respectively.

The dominant improvement is obtained by the proposed compact pipeline, which propagates the compact valid-path representation through the field-calculation stage. In this final implementation, field calculation directly processes the compact valid-path records instead of revisiting full candidate or path-length buffers. For $R = 6, D = 1$, the total runtime decreases from 3725.4 s to 2563.0 s, corresponding to a 31.2% reduction. For $R = 6, D = 2$, the total runtime decreases from 3794.6 s to 2693.4 s, corresponding to a 29.0% reduction. The proposed compact pipeline also reduces the peak CPU working set from 18.3 GB to 16.2 GB for $R = 6, D = 1$ and from 26.7 GB to 24.7 GB for $R = 6, D = 2$. The peak GPU memory remains 22.9 GB after the compact-transfer stage, indicating that the final compact pipeline improves how valid-path data are consumed and retained without increasing the peak GPU memory footprint beyond the compact-buffer allocation.

Table I(b) compares the final compact-pipeline implementation for $R = 6, D = 1$ and $R = 6, D = 2$. In this benchmark, enabling double diffraction increases the total runtime from 2563.0 s to 2693.4 s, corresponding to an additional 130.4 s or 5.1%. This overhead is mainly concentrated in VT construction and field calculation, which increase by 63.4 s and 35.3 s, respectively. In contrast, the PT+ST time remains essentially unchanged, and the peak GPU memory stays at 22.9 GB for both configurations. Although the peak CPU working set increases by 8.5 GB when moving from $D = 1$ to $D = 2$, this increase is consistent with the larger host-side state required to store double-diffraction path candidates and associated metadata.

These results show that the proposed compact valid-path pipeline controls the additional cost of double diffraction in the evaluated large-scale urban scenario, with the remaining overhead concentrated in the stages that explicitly generate and evaluate additional double-diffraction paths, particularly VT construction and field calculation. The final $R = 6, D = 2$ compact-pipeline result, with a total runtime of 2693.4 s and a peak GPU memory usage of 22.9 GB, is therefore used as the benchmark performance of the proposed MD-integrated IT-RT engine. Further acceleration of the special-function evaluations required in the field-calculation stage, such as repeated Fresnel-integral and transition-function computations, remains an active research topic and is left for future work [32], [42]–[44]. Nevertheless, by combining DABS-based memory-efficient VT construction, GPU-accelerated PT/ST, and the proposed compact valid-path pipeline, the developed engine enables unprecedented urban-scale IT-RT simulations with full 3-D multi-diffraction over arbitrary 3-D wedge geometries.

V. COMPARISON WITH SIONNA RT IN AN URBAN CAD SCENARIO

Recent benchmarks of SBR-RT simulators show that shadow-region predictions can differ across simulators, as illustrated in [24, Fig. 9]. This difference is associated with the ray-generation characteristics of SBR-type ray launching, including finite angular sampling and cutoff rules, under which some geometrically valid paths may not be sampled

or recovered [3]. Accordingly, comparing IT-RT with SBR-RT provides a useful validation reference for RT studies and practical simulator development [9], [10], [12]. Related RT simulators and their supported interaction capabilities are summarized in [3, Table III]. For a fair comparison, Sionna RT and the proposed IT-RT engine used the same urban-scale CAD scenario and Tx/Rx configuration as those described in Section IV-E. Sionna RT was configured with the largest ray-count setting, namely `ray_depth=5`, `refraction=false`, `edge_diffraction=true`, and `samples_per_src=106`. Under this setting, Sionna completed the simulation in 272.694 s with a peak resident set size of 1067.4 MiB.

For the proposed engine, the final $R = 6, D = 2$ configuration generated 1,021,242 total paths. Among them, 635,823 paths were multi-diffraction paths, accounting for more than 60% of all computed paths. In contrast, Sionna with `ray_depth=5` identified 94,581 total rays under the same Tx/Rx condition. Thus, although the Sionna run completed with a relatively small runtime and memory footprint, its ray set contained only about 9.3% of the total number of paths obtained by the proposed engine and fewer rays than the multi-diffraction paths alone in the proposed result.

Fig. 12(a) shows the Rx locations where the proposed IT-RT engine resolves valid propagation while Sionna produces missing-ray results. This observation is consistent with the known missing-path behavior in ray-launching and SBR-RT methods [9], [10], [12], [18]. In particular, the missing-ray regions frequently appear near or behind blockage-rich CAD building footprints and at larger Tx–Rx distances, where multi-bounce and diffraction-assisted paths are more likely to contribute to the received field.

Fig. 12(b) quantifies this distance dependence using the distance-binned Sionna missing-ray ratio. For each Tx–Rx distance bin, this ratio is defined as the fraction of proposed-IT-RT-valid Rx locations for which Sionna produces missing-ray results. The ratio increases from 0.0% within 50 m to 46.1% in the 350–400 m bin, indicating that Sionna missing-ray cases become more pronounced in farther Rx regions. This distance-dependent trend is also supported by the overall Rx-distance statistics: among the proposed-IT-RT-valid Rx locations, the median Tx–Rx distance is 177.6 m for the Sionna-valid group and 253.9 m for the Sionna-missing group. These results indicate that Sionna missing support is concentrated in long-range and geometrically complex propagation regions.

If these paths are not recovered, the received-field calculation can omit valid propagation contributions, leading to path-loss overestimation or artificial shadowed regions in the coverage map. Such effects are especially important for 6G upper-mid-band coverage analysis, where accurate cell-edge prediction is needed to assess the additional link budget, beamforming gain, or deployment densification required to approach 5G-comparable coverage [2], [3]. Therefore, the observed difference in this urban scenario can serve as a useful reference for improving the accuracy and coverage reliability of practical RT simulators.

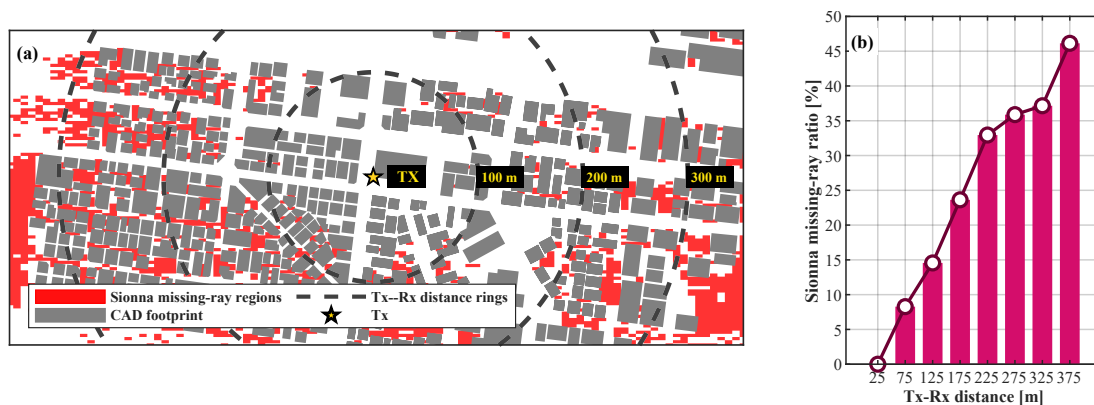


Fig. 12. Propagation-support comparison between the proposed IT-RT engine and Sionna RT. (a) Rx locations where the proposed engine resolves valid propagation while Sionna ray-depth 5 produces missing-ray results. Red regions denote proposed-valid/Sionna-missing Rx locations, gray polygons denote CAD footprints, and dashed curves denote Tx-Rx distance rings. (b) Distance-binned Sionna missing-ray ratio over the proposed-valid Rx set.

VI. CONCLUSION AND FUTURE WORK

This work developed a full 3-D, IT-based, CPU/GPU-accelerated RT framework that explicitly integrates multi-diffraction over arbitrary urban wedge configurations. The implemented MD solver was validated from path tracing to field evaluation: the traced double-diffraction paths satisfied Keller's law, and the second-order UTD field computation resolved transition-region discontinuities that remain in a single-diffraction-only engine.

To improve robustness in complex urban CAD geometries, we introduced the SGN optimizer for singularity-sensitive double-diffraction configurations near the $Q_1 \approx Q_2$ collapse region. SGN prevented premature termination near the non-differentiable collapse set while preserving the fast C-A convergence behavior for ordinary differentiable paths. For scalable execution, SGN was reorganized using a device-compact split structure, and the resulting robust MD solver was integrated into the full IT-RT engine with DABS-based visibility-tree construction, NVIDIA OptiX-based path tracing/shadowing, and the compact valid-path pipeline.

In the urban CAD benchmark, the proposed compact pipeline reduced the $R = 6, D = 2$ runtime from 3794.6 s to 2693.4 s and limited the additional double-diffraction overhead to 130.4 s, or 5.1%. The comparison with Sionna RT further showed that the proposed engine provides richer propagation support in blockage-rich and long-distance regions where Sionna frequently produces missing-ray results. These results indicate that robust full-3-D MD-aware IT-RT can serve as a useful reference framework for improving the accuracy, completeness, and coverage reliability of practical RT simulators for 6G upper-mid-band studies. Future work will pursue faster field calculation, extended propagation mechanisms such as transmission and diffuse scattering, and AI-assisted IT-RT acceleration for 6G coverage and system-level studies.

REFERENCES

- [1] H. Miao, J. Zhang, P. Tang, L. Tian, X. Zhao, B. Guo, and G. Liu, "Sub-6 ghz to mmwave for 5g-advanced and beyond: Channel measurements, characteristics and impact on system performance," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 6, pp. 1945–1960, Jun. 2023.
- [2] H. Yang, S. Bang, H. Kim, S. Kim, Y. Sung, K. Park, S. Kim, and J. Oh, "6g channel measurement in urban, dense urban scenario and massive ray-tracing-based coverage analysis at 7.5 ghz," in *ICC 2025 - IEEE International Conference on Communications*, Jun. 2025, pp. 3309–3314.
- [3] H. Yang, H. Kim, U. Park, S. Bang, S. Kim, Y. Sung, K. Park, S. Kim, and J. Oh, "6g upper mid-band channels at 7.5-ghz with diverse urban campaigns and coverage analysis via image-theory full-3-d ray-tracing," *IEEE Trans. Commun.*, vol. 74, no. 12, pp. 9802–9817, Dec. 2026.
- [4] F. Capolino, I. Albani, S. Maci, and R. Tiberio, "Double diffraction at a pair of coplanar skew edges," *IEEE Trans. Antennas Propag.*, vol. 45, no. 8, pp. 1219–1226, Aug. 1997.
- [5] P. Holm, "A new heuristic utd diffraction coefficient for nonperfectly conducting wedges," *IEEE Trans. Antennas Propag.*, vol. 48, no. 8, pp. 1211–1219, Aug. 2000.
- [6] M. Schneider and R. Luebbers, "A general, uniform double wedge diffraction coefficient," *IEEE Trans. Antennas Propag.*, vol. 39, no. 1, pp. 8–14, Jan. 1991.
- [7] M. Albani, "A uniform double diffraction coefficient for a pair of wedges in arbitrary configuration," *IEEE Trans. Antennas Propag.*, vol. 53, no. 2, pp. 702–710, Feb. 2005.
- [8] G. Carluccio, F. Puggelli, and M. Albani, "Generalization of utd double diffraction to the case of impenetrable wedges with relatively general boundary conditions," *IEEE Trans. Antennas Propag.*, vol. 62, no. 7, pp. 3829–3834, Jul. 2014.
- [9] C. Key, B. A. Troksa, S. Kasdorf, and B. M. Notaroš, "Non-self-adjacent ray classes for parallelizable shooting-bouncing ray tracing double count removal," *IEEE J. Multiscale and Multiphys. Comput. Techn.*, vol. 5, pp. 245–254, 2020.
- [10] S. Kasdorf, B. Troksa, C. Key, J. Harmon, and B. M. Notaroš, "Advancing accuracy of shooting and bouncing rays method for ray-tracing propagation modeling based on novel approaches to ray cone angle calculation," *IEEE Trans. Antennas Propag.*, vol. 69, no. 8, pp. 4808–4815, Aug. 2021.
- [11] M. Lecci, P. Testolina, M. Polese, M. Giordani, and M. Zorzi, "Accuracy versus complexity for mmwave ray-tracing: A full stack perspective," *IEEE Trans. Wireless Commun.*, vol. 20, no. 12, pp. 7826–7841, Dec. 2021.
- [12] S. Kasdorf, B. Troksa, C. Key, J. Harmon, S. Pasricha, and B. M. Notaroš, "Parallel gpu optimization of the shooting and bouncing ray tracing methodology for propagation modeling," *IEEE Trans. Antennas Propag.*, vol. 72, no. 1, pp. 174–182, Jan. 2024.
- [13] K. Yang, C. Liu, W. Yu, and T. Jun Cui, "Pointemray: A novel efficient sbr framework on point-based geometry," *IEEE Trans. Antennas Propag.*, vol. 73, no. 1, pp. 375–390, Jan. 2025.
- [14] J.-H. Lee, J.-S. Choi, and S.-C. Kim, "Cell coverage analysis of 28 ghz millimeter wave in urban microcell environment using 3-d ray tracing," *IEEE Trans. Antennas Propag.*, vol. 66, no. 3, pp. 1479–1487, Mar. 2018.
- [15] F. Puggelli, G. Carluccio, and M. Albani, "An efficient algorithm for the computation of the utd t transition function," *IEEE Trans. Antennas Propag.*, vol. 60, no. 5, pp. 2380–2387, May 2012.

- [16] G. Carluccio, F. Puggelli, and M. Albani, "Algorithm for the computation of the generalized fresnel integral," *IEEE Trans. Antennas Propag.*, vol. 59, no. 10, pp. 3943–3947, Oct. 2011.
- [17] G. Carluccio and M. Albani, "An efficient ray tracing algorithm for multiple straight wedge diffraction," *IEEE Trans. Antennas Propag.*, vol. 56, no. 11, pp. 3534–3542, Nov. 2008.
- [18] J. Tan, Z. Su, and Y. Long, "A full 3-d gpu-based beam-tracing method for complex indoor environments propagation modeling," *IEEE Trans. Antennas Propag.*, vol. 63, no. 6, pp. 2705–2718, Jun. 2015.
- [19] M. F. Catedra, J. Perez, F. S. De Adana, and O. Gutierrez, "Efficient ray-tracing techniques for three-dimensional analyses of propagation in mobile communications: application to picocell and microcell scenarios," *IEEE Antennas Propag. Mag.*, vol. 40, no. 2, pp. 15–28, Apr. 1998.
- [20] L. Jingyu, Y. Wang, J. Huang, Y. Li, J. Huang, and C.-X. Wang, "An improved triangular facets based angular z-buffer algorithm for im ray tracing channel modeling," in *2022 IEEE/CIC International Conference on Communications in China (ICCC)*. IEEE, Aug. 2022, pp. 1050–1056.
- [21] C. Saeidi and F. Hodjatkishani, "Modified angular z-buffer as an acceleration technique for ray tracing," *IEEE Trans. Antennas Propag.*, vol. 58, no. 5, pp. 1822–1825, May 2010.
- [22] K. Rizk, J.-F. Wagen, and F. Gardiol, "Two-dimensional ray-tracing modeling for propagation prediction in microcellular environments," *IEEE Trans. Veh. Technol.*, vol. 46, no. 2, pp. 508–518, May 1997.
- [23] G. E. Athanasiadou, A. R. Nix, and J. P. McGeehan, "A microcellular ray-tracing propagation model and evaluation of its narrow-band and wide-band predictions," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 3, pp. 322–335, Mar. 2000.
- [24] M. Zhu, L. Cazzella, F. Linsalata, M. Magarini, M. Matteucci, and U. Spagnolini, "Toward real-time digital twins of EM environments: Computational benchmark for ray launching software," *IEEE Open J. Commun. Soc.*, vol. 5, pp. 6291–6302, Sep. 2024.
- [25] M. G. Aram, H. Guo, M. Yin, and T. Svensson, "Site-specific outdoor propagation assessment and ray-tracing analysis for wireless digital twins," in *2025 19th European Conference on Antennas and Propagation (EuCAP)*. Stockholm, Sweden: IEEE, Mar. 2025, pp. 1–5.
- [26] F. A. Aoudia, J. Hoydis, M. Nimier-David, S. Cammerer, and A. Keller, "Sionna RT: Technical report," *CoRR*, vol. abs/2504.21719, Apr. 2025.
- [27] *Release Notes: Altair Feko 2025.1*, 2025, Accessed: Nov. 7, 2025. [Online]. Available: https://help.altair.com/Winprop/topics/feko/release_notes/2025_1/release_notes_intro_feko_winprop_r.htm
- [28] H. Yang, S. Kim, H. Kim, S. Bang, Y. Kim, S. Kim, K. Park, D. Kwon, and J. Oh, "Beyond limitations of 5g with ris: Field trial in a commercial network, recent advances, and future directions," *IEEE Commun. Mag.*, vol. 62, no. 10, pp. 132–138, Oct. 2024.
- [29] J. Gomez-Ponce, D. Chizhik, J. Du, R. Valenzuela, K. Arana, N. A. Abbasi, and A. F. Molisch, "Efficient integration of street clutter into mm-wave ray tracing in urban environments based on more than 1000 measurements," *IEEE Trans. Antennas Propag.*, vol. 74, no. 1, pp. 1257–1262, Jan. 2026.
- [30] A. Bazzi, M. Ying, O. Kanhere, T. S. Rappaport, and M. Chafii, "ISAC imaging by channel state information using ray tracing for next generation 6G," *IEEE J. Sel. Top. Electromagn. Antennas Propag.*, vol. 1, no. 1, pp. 135–149, Sep. 2025.
- [31] M. Abramowitz and I. A. Stegun, *Handbook of mathematical functions: with formulas, graphs, and mathematical tables*. Courier Corporation, 1965, vol. 55.
- [32] Ö. Işık, M. E. Köksal, and A. Uzer, "Communication evaluation of the remainders of a class of series by using saddle-point method," *IEEE Trans. Antennas Propag.*, vol. 71, no. 2, pp. 2011–2015, Feb. 2023.
- [33] X. Chen, "Smoothing methods for nonsmooth, nonconvex minimization," *Math. Program.*, vol. 134, no. 1, pp. 71–99, Aug. 2012.
- [34] A. Y. Aravkin, R. Baraldi, and D. Orban, "A proximal quasi-newton trust-region method for nonsmooth regularized optimization," *SIAM J. Optim.*, vol. 32, no. 2, pp. 900–929, Jun. 2022.
- [35] J.-y. Fan and Y.-x. Yuan, "On the quadratic convergence of the levenberg-marquardt method without nonsingularity assumption," *Computing*, vol. 74, no. 1, pp. 23–39, Feb. 2005.
- [36] T. Dehghan Niri, M. Heydari, and M. Hosseini, "Correction of trust region method with a new modified newton method," *Int. J. Comput. Math.*, vol. 97, no. 5, pp. 1118–1132, May 2020.
- [37] Blender Online Community, *Blender—a 3-D Modelling and Rendering Package*, Stichting Blender Foundation, Amsterdam, 2018, Accessed: Nov. 7, 2025. [Online]. Available: <http://www.blender.org>
- [38] OpenStreetMap contributors, "About OpenStreetMap," 2025, Accessed: Nov. 7, 2025. [Online]. Available: https://wiki.openstreetmap.org/wiki/About_OpenStreetMap
- [39] *V-World*, 2025, Accessed: Nov. 7, 2025. [Online]. Available: <https://map.vworld.kr/map/dtkmap.do?mode=MAPW201>
- [40] *NVIDIA OptiX 7.7—Programming Guide*, 2025, Accessed: Nov. 7, 2025. [Online]. Available: <https://raytracing-docs.nvidia.com/optix7/guide/index.html#preface#>
- [41] Y. Kim, H. Yang, H. Kim, J. Jo, and J. Oh, "Anxel beam shrinkage method and heterogeneous computing-accelerated full-image theory method ray tracing enabling massive outdoor propagation modeling," *IEEE Trans. Antennas Propag.*, vol. 72, no. 7, pp. 5935–5949, Jul. 2024.
- [42] A. Uzer, "Evaluations of a class of integrals that arise in wave scattering problems by using the taylor product theorem," *IEEE Trans. Antennas Propag.*, vol. 66, no. 2, pp. 1031–1034, Feb. 2018.
- [43] —, "Series expansions immune to round-off errors for the transition function used in high frequency electromagnetic methods," *IEEE Antennas Wireless Propag. Lett.*, vol. 20, no. 7, pp. 1279–1282, Jul. 2021.
- [44] —, "Techniques for obtaining non-divergent series as alternatives to divergent asymptotic series for special functions and integrals," *IEEE Trans. Antennas Propag.*, vol. 73, no. 9, pp. 6627–6634, Sep. 2025.



Hyunjun Yang received a B.S. degree in electronic and electrical engineering from Pohang University of Science and Technology (POSTECH), Pohang, South Korea, in 2020. He is currently pursuing an integrated master's and Ph.D. degree in the Department of Electrical Engineering and Computer Science at Seoul National University, Seoul, South Korea. His current research interests include ray-tracing EM analysis techniques, EM theory, wireless propagation channel measurement & modeling, and reconfigurable intelligent surfaces (RIS).



Hooyoung Kim received the B.S. degree in electrical and computer engineering from Seoul National University, Seoul, Korea, in 2024. He is currently pursuing an integrated master's and Ph.D. degree in the Department of Electrical and Computer Engineering at Seoul National University, Seoul, Korea. His current research interests include ray-tracing EM analysis technique for 6G communication, wireless propagation channel measurement and AI-enabled EM analysis.



Jungsuek Oh (S'08) received B.S. and M.S. degrees from Seoul National University, Korea, in 2002 and 2007, respectively, and a Ph.D. degree from the University of Michigan at Ann Arbor, in 2012. From 2007 to 2008, he was with Korea Telecom as a Hardware Research Engineer, working on the development of flexible RF devices. In 2012, he was a Postdoctoral Research Fellow in the Radiation Laboratory at the University of Michigan. From 2013 to 2014, he was a Staff RF Engineer with Samsung Research America, Dallas, working as a project leader for the 5G/millimeter-wave antenna system. From 2015 to 2018, he was a faculty member in the Department of Electronic Engineering at Inha University in South Korea. He is currently an Professor in the School of Electrical and Computer Engineering at Seoul National University in South Korea. His research areas include mmWave beam focusing/shaping techniques, antenna miniaturization for integrated systems, and radio propagation modeling for indoor scenarios. He is the recipient of the 2011 Rackham Predoctoral Fellowship Award at the University of Michigan. He has published more than 40 technical journal and conference papers. He has served as a technical reviewer for IEEE Transactions on Antennas and Propagation and IEEE Antenna and Wireless Propagation Letters, among other journals. He has served as a TPC member and as a session chair for the IEEE AP-S/USNC-URSI and ISAP. He has been a senior member of IEEE since 2017.