

Monte Carlo Methods for Linear Elliptic Boundary Value Problems

Bailey Mark Miller

CMU-CS-26-129

August 2026

Computer Science Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Ioannis Gkioulekas (Chair)
Keenan Crane
Nicholas Boffi

Ravi Ramamoorthi (UC San Diego)
Mathieu Desbrun (Inria and École Polytechnique)

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.*

Copyright © 2026 Bailey Mark Miller

This research was sponsored by National Science Foundation awards 1900849, 1943123, 2008123, 2047341, and 2212290; National Institute of Food and Agriculture award 2023-67021-39073; gifts from nTopology, Adobe Research, and Disney Research; a National Science Foundation Graduate Research Fellowship under Grant Nos. DGE1745016 and DGE2140739, an NVIDIA Graduate Fellowship, and a Carnegie Mellon Presidential Fellowship supported by Tata Consultancy Services; a Packard Foundation Fellowship; and Alfred P. Sloan Research Fellowship FG202013153.

The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: Monte Carlo methods, partial differential equations, elliptic boundary value problems, geometry processing, grid-free simulation, walk on spheres, walk on stars

To the Millerbillies

Abstract

This thesis develops the walk on spheres family of Monte Carlo PDE solvers into a practical computational framework for solving linear elliptic boundary value problems. Walk on spheres is effective for the same reason Monte Carlo rendering is: both recast their governing equations—elliptic PDEs and light transport—as integral equations that admit recursive Monte Carlo estimators, and both inherit the geometric scalability and robustness that follow from sampling rather than meshing. Yet where rendering matured over the past three decades into a ubiquitous, practical tool, walk on spheres still lacks the corresponding building blocks: support for more general boundary conditions, generalizations to participating media, caching and reuse schemes for accelerated evaluation, and differentiable variants for shape optimization. We take direct inspiration from the methods in rendering that provide these capabilities.

We generalize walk on spheres to first-order linear boundary conditions, broadening it beyond pure Dirichlet problems to the wider range of physically meaningful boundary models, much as general reflectance models did for rendering. We extend the method to participating media, solving linear elliptic boundary value problems on the same intricate microparticle geometries that volume rendering handles. We introduce caching and reuse schemes, in the spirit of virtual point lights and irradiance caching, that produce efficient and smooth dense solution estimates. Finally, we develop differential walk on spheres, computing solution derivatives through a nested boundary value problem that walk on spheres solves recursively, in analogy to differentiable rendering. Together, these contributions elevate walk on spheres from a geometrically capable estimator into a practical and extensible framework for Monte Carlo PDE simulation, mirroring the trajectory of Monte Carlo rendering.

A Note About Content

The text and figures in this thesis have significant overlap with several papers published by myself and collaborators over the course of my PhD. These publications are the original source for such content, used with permission.

1. Rohan Sawhney*, Bailey Miller*, Ioannis Gkioulekas[†], and Keenan Crane[†]. Walk on Stars: A Grid-Free Monte Carlo Method for PDEs with Neumann Boundary Conditions. *ACM Transactions on Graphics* (2023).
2. Bailey Miller*, Rohan Sawhney*, Keenan Crane[†], and Ioannis Gkioulekas[†]. Boundary Value Caching for Walk on Spheres. *ACM Transactions on Graphics* (2023).
3. Bailey Miller*, Rohan Sawhney*, Keenan Crane[†], and Ioannis Gkioulekas[†]. Walkin’ Robin: Walk on Stars with Robin Boundary Conditions. *ACM Transactions on Graphics* (2024).
4. Bailey Miller, Rohan Sawhney, Keenan Crane, and Ioannis Gkioulekas. Differential Walk on Spheres. *ACM Transactions on Graphics* (2024).
5. Bailey Miller, Rohan Sawhney, Keenan Crane, and Ioannis Gkioulekas. Solving Partial Differential Equations in Participating Media. *ACM Transactions on Graphics* (2025).

The symbols * and [†] denote equal contribution among the marked authors.

Parts of this thesis additionally draw on material developed for two courses: *State of the Art in Grid-Free Monte Carlo Methods for Partial Differential Equations* [Sawhney et al. 2025], by Rohan Sawhney, Bailey Miller, Ioannis Gkioulekas, and Keenan Crane; and *Vision, Imaging, and Simulation for Heat and Light* [Ramanagopal and Miller 2025], by Mani Ramanagopal and Bailey Miller.

Acknowledgments

This thesis, and the journey leading up to it, would not have been possible without the guidance, support, and friendship of so many people. I am grateful to have spent some of the most challenging and joyous years of my life working and living alongside exceptionally talented and kind souls.

First and foremost, I would like to thank my advisor Ioannis Gkioulekas for his mentorship and for teaching me to think carefully, lean into the math, and insist on rigor. His vast command of so many topics brought a unifying clarity to everything we worked on. Some of the most formative parts of my PhD were spent exploring problems together at the whiteboard, consistently running well past our scheduled meeting time—my apologies to anyone whose meeting followed ours. I would also like to thank Keenan Crane, my unofficial second advisor, who taught me to see the simple elegance in so many areas of applied math, and who always brought the best nerd-sniping problems to Smith Hall lunches. Finally, many thanks to Rohan Sawhney, whose mentorship and collaboration played a pivotal role at the start of my PhD. I could not have asked for a better group than Ioannis, Keenan, and Rohan with whom to pursue an ambitious research agenda. Thank you as well to my thesis committee, including Nicholas Boffi, Ravi Ramamoorthi, and Mathieu Desbrun, for their thoughtful feedback and engagement with this work.

I am also grateful to the many collaborators I was fortunate to work with at Carnegie Mellon and during internships, including Hanyu Chen, Alice Lai, Tanli Su, Kevin Mu, Zachary Tatlock, Steven Lee, Mani Ramanagopal, Miloš Hašan, Krishna Mullia, Valentin Deschaintre, Nathan Carr, Lampros Mountrakis, Fabrice Rousselle, Jan Novák, Eugene d'Eon, Lifan Wu, Zihan Yu, Shuang Zhao, Desmond Liu, Bernd Bickel, Delio Vicini, Bharath Seshadri, and Georgios Kopanas.

Throughout my time at Carnegie Mellon, Smith Hall was a second home, and the people in it made it so. Thank you to Arjun Teh, Mark Gillespie, Arjun Lakshmipathy, Hossein Baktash, Nicole Feng, Olga Gutan, Sreekar Ranganathan, Zoë Marschner, Yuichi Hirose, Mian Wei, and Ruslana Fogler. Outside of Smith Hall, thank you to Saranya Vijayakumar, Alexander Goldberg, Bernardo Subercaseaux, and Pascal for friendship and for being my sounding boards, often over a glass of wine.

Before Carnegie Mellon, I would like to thank Wojciech Jarosz for introducing me to computer graphics research. Wojciech's rendering algorithms course opened my eyes to Monte Carlo methods and computer graphics more broadly. It led to my very first research experience—three years of work that set me on the path to graduate school. Thank you also to Gurprit Singh and Iliyan Georgiev for their mentorship during those early projects.

To my family, thank you for everything. I would especially like to thank my parents, John and Lisa Miller, for their lifelong love, support, and encouragement. To my older brother and best friend John: thank you for being my biggest advocate, for listening to me rant endlessly about my research, and for letting me spend several weeks on your couch as I completed this thesis and figured out life after grad school.

Finally, my deepest thanks go to my partner Tsui Yun Chang, for unwavering support and boundless happiness through the latter half of this PhD. The impromptu care packages sent during my job search and thesis writing sustained me. Now that this journey is over, I can't wait for the rest of ours to begin.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Approach and Scope	4
1.3	Contributions	5
2	Background	7
2.1	Monte Carlo Estimation	7
2.2	Volterra Integral Equations	8
2.3	PDEs as Integral Equations	9
2.4	The Walk on Boundary Alternative	10
2.5	Poisson Boolean Model	12
2.6	A Tour of Monte Carlo Rendering	14
3	First-Order Linear Boundary Conditions	17
3.1	Recursive Boundary Integral Equation	19
3.2	Walk on Stars	20
3.3	Operator-Theoretic Convergence	24
3.4	Evaluation	25
3.5	Outlook	27
4	Participating Media	29
4.1	Boundary Value Problems in Participating Media	31
4.2	Volumetric Walk on Spheres	32
4.3	Conditional Closest-Point Sampling	34
4.4	Volumetric Walk on Stars	36
4.5	Evaluation	37
4.6	Outlook	40
5	Caching-Based Acceleration	42
5.1	Boundary Integral Equation and Sample Reuse	43
5.2	Practical Considerations	45
5.3	Evaluation	48
5.4	Outlook	48

6	Differentiable Simulation	50
6.1	Differential Boundary Value Problems	52
6.2	First-Order Linear Boundary Conditions	54
6.3	Differential Walk on Spheres	55
6.4	Evaluation	57
6.5	Outlook	59
7	Future Directions	61
7.1	DSLs for Recursive Integral Equations	62
7.2	Stochastic Geometry $\times$ Monte Carlo PDE Solvers	62
7.3	Neural Fields $\times$ Monte Carlo PDE Solvers	63
7.4	Closing Remarks	64

List of Figures

1.1	The walk on stars (WoSt) family of estimators developed in this thesis enables a richer class of problems than the original walk on spheres (WoS) method. Here, for instance, we simulate diffusive convective heat transfer from a toaster (Dirichlet) to a piece of bread (Neumann) by solving a Laplace equation with mixed boundary conditions (top and bottom right), complementing the radiative transfer computed via ray tracing (bottom left). Just like ray tracing, we can simulate directly on the full high-resolution data (bottom center) without generating a volume mesh or forming a global stiffness matrix. Since results are progressive, we can preview how the toast will look faster than it takes to toast a real piece of bread (top left).	1
1.2	<i>(Left)</i> A non-negative Robin coefficient μ interpolates between reflecting Neumann ($\mu = 0$) and absorbing Dirichlet ($\mu \rightarrow \infty$) conditions, which prescribe boundary derivatives and values, respectively. As μ increases and the boundary becomes less reflecting, Russian roulette probabilistically terminates walks on the Robin boundary, reducing the average number of steps per walk. <i>(Right)</i> The rendering analogue: a room with perfectly specular surfaces reflects all incident radiance, like a Neumann boundary; realistic scenes include materials that only partially reflect; and purely emissive surfaces behave like prescribed Dirichlet values. As in rendering, richer material models yield more useful and realistic solutions.	3
1.3	<i>(Left)</i> Generating tetrahedral meshes for accurate FEM simulation is challenging on detailed models: state-of-the-art tools either fail to capture important detail or run out of memory at finer tolerances—here, fTetWild [Hu et al. 2020] on the Mars rover of Figure 3.1 with 64 GB of memory, while TetGen [Si 2015] cannot tetrahedralize the model at all due to self-intersections. <i>(Right)</i> Even for simple boundary conditions, boundary element methods suffer large global errors without significant mesh refinement, due to local aliasing of boundary data, and can fail outright on irregular elements (top row).	4
1.4	Where will oxygen flow at the beginning of a breath? Walk on stars simulates gas exchange in a lung model whose thin features defeat volumetric meshing—the kind of problem the contributions of this thesis make accessible: general boundary conditions, evaluated directly on complex geometry (Chapter 3).	6

- 3.1 Thermal analysis of NASA’s Curiosity rover with walk on stars (WoSt): a Laplace equation with Robin boundary conditions solved directly on an artist-created model that state-of-the-art tetrahedral meshers cannot robustly process. Deferred shading launches walks only at visible points, with sample density proportional to screen coverage (Section 3.4). 17
- 3.2 (*Left*) Selecting the star radius. A naïve radius capped only by the absorbing boundary admits occluded points, where the Poisson kernel is no longer a valid density; restricting to the closest silhouette restores a star-shaped region ($\mu = 0$), and for $\mu > 0$ the radius shrinks further to keep $\rho_\mu \in [0, 1]$ —as $\mu \rightarrow \infty$ it collapses to the boundary distance, recovering absorbing behavior. (*Right*) The resulting estimator handles arbitrary mixtures of boundary conditions with a single walk: walks land on reflecting and partially reflecting boundaries and continue, surviving Russian roulette with probability ρ_μ , and terminate in the ε -shell of absorbing boundaries. 22
- 3.3 A single ε tolerance controls the tradeoff between bias and efficiency. On predominantly reflecting Robin boundaries, overly large ε manifests as a global darkening of the estimated solution (top two rows); on absorbing boundaries it instead produces Voronoi-like flattening (bottom row). 23
- 3.4 (*Left*) Prior methods simulate reflections by nudging walks back into the domain by a fixed offset ζ , so walks take many small steps near reflecting boundaries. (*Right*) At comparable parameter settings, WoSt is both faster and less biased than WoS with discretized reflections or estimators based on stochastic differential equations (SDEs): where discretized walks plateau at a bias floor—ignoring geometry below their offset scale—WoSt continues to converge at the expected Monte Carlo rate. 25
- 3.5 Walk on boundary suffers an extreme bias–variance tradeoff in nonconvex domains: truncating walks introduces severe bias, while untruncated walks exhibit exponential variance growth. WoSt instead converges reliably at the expected Monte Carlo rate for any combination of Dirichlet, Neumann, and Robin conditions. 26
- 4.1 Physical systems such as biological membranes involve extraordinarily complex microparticle geometry—here, ions in solution mediating electrostatic interactions. Volumetric walk on spheres estimates the mean solution of a PDE over such geometry while realizing only the particles each walk encounters (Section 4.5). 29
- 4.2 (*Left*) We consider domains $\Omega_O := V \setminus O$, equal to the difference between a deterministic volume V and a random configuration of particles O contained in the volume, drawn from a Poisson Boolean model whose density λ is defined over the volume. (*Right*) Ensemble averaging is a simple but expensive way to estimate the mean solution: sample many particle configurations (top), solve the PDE on each realized domain (bottom), and average. Volumetric walk on spheres instead estimates the mean solution directly. 31

- 4.3 (Left) WoS jumps along spheres whose radii are set by deterministic closest-point queries; volumetric WoS jumps the same way, but samples closest points from the unresolved microparticle geometry, realizing only the particles each walk encounters. (Center right) With absorbing particles, sampling the closest center $c(x)$ determines both the closest point $q_{\partial O}^*(x)$ and its distance $r_{\partial O}(x)$ —the radius of the next empty ball. (Right) With reflecting particles, the star-shaped region St is bounded instead by the nearer of the deterministic boundary and the closest particle silhouette $q_{\text{sil},O}^*(x)$ 32
- 4.4 (Left) The unconditional closest-point sampler: in a homogeneous medium, an exponential variate determines the distance to the closest particle center; in a heterogeneous medium, thinning fills the volume with fictitious density and rejects candidates until a real particle is accepted. The sampled distance follows the contact distribution p_x^{dc} . (Right) Conditional sampling during a walk: (a) each step compares the sampled stochastic candidate $q_{\partial O}^*$ against the deterministic candidate $q_{\partial V}^*$ and previously realized geometry, taking the nearer point q^* ; (b, c) the memory of empty regions and sampled particles zeroes the density where the medium has already been observed, keeping later samples consistent with earlier ones. 34
- 4.5 (Left) Mean solutions with reflecting particles: across more and less concentrated media, VWoSt matches ensemble-averaged references up to Monte Carlo error while being nearly $5\times$ faster. (Right) The need for memory: truncating the walk's memory to finite size improves runtime marginally or not at all, yet always introduces significant bias, across both absorbing and reflecting particles. 38
- 4.6 Compared to a reference mean solution from ensemble averaging, VWoS produces unbiased estimates across particle sizes, whereas homogenization develops visible bias as particle radius grows—largest near thin geometric features and the medium boundary, where the particle scale is no longer negligible. 39
- 4.7 Coupled Monte Carlo simulation of cloud photochemistry: primary rays trace sunlight to its first collision inside the cloud, where the absorbed energy generates ozone, and VWoS estimates the diffusion of the resulting chemical species. Both estimators operate on the same Poisson Boolean medium, and sharing the walk's memory keeps their estimates consistent, capturing shadowing effects lost under independent sampling. 40
- 5.1 Streamlines traced through a simulated wind tunnel. Because the gradient comes from the same cached boundary samples as the solution, a single cached solve traces the entire flow; at equal time, cached streamlines are smoother and better converged than pointwise estimation (Section 5.3). 42
- 5.2 (Left) The free-space Green's function and its normal derivative are singular at the evaluation point but decay smoothly and fall off quickly away from it. (Right) Because cached samples are shared rather than importance sampled, evaluation points near the boundary exhibit singular artifacts (unclamped); clamping the kernels suppresses the artifacts but biases the solution, while the correction step removes them without bias. 45

- 5.3 Cached reconstructions exhibit correlated rather than pointwise error: with few boundary samples the estimated solution is smooth but globally inaccurate, and this coherent global error vanishes as boundary samples accumulate. 47
- 5.4 Thermal regulation of a spacesuit, with ray-traced solar radiation supplying Dirichlet data, outflow modeled by Neumann conditions, and an interior source for the astronaut's body heat. At equal time, the cached solution on a dense slice is markedly smoother than pointwise estimation. 47
- 5.5 (*Left*) Caching yields far smoother results across the domain than pointwise estimation at equal time, and evaluating the boundary integral at more interior points has little impact on performance, since the same boundary samples determine the solution everywhere. (*Right*) Unlike a global solve involving the entire boundary, computation can focus on a region of interest by caching points only on the boundary of a smaller subdomain. 48
- 6.1 Differential walk on spheres recovers the shape of an emissive object from its diffusion profile, observed on a box surrounding the object, via gradient-based optimization. Unlike mesh- or grid-based approaches, derivatives are evaluated only at the points of interest—here, the observation locations on the box—without ever computing a global solution (Section 6.4). 50
- 6.2 (*Left*) Derivatives can be computed on a wide range of boundary representations, including those not directly handled by conventional solvers—implicit radial basis functions, splines, and meshes. For a solution $u(x, \pi)$ (middle row), the derivative $\dot{u}(x, \pi)$ (bottom row) is taken with respect to positional parameters π of the boundary $\partial\Omega(\pi)$. (*Right*) Differential walk on spheres computes these derivatives. Like ordinary walk on spheres (WoS), the differential walk recursively jumps to a random point on the largest empty sphere until it terminates in the ε -shell—but there, instead of evaluating prescribed boundary data, it launches a primary walk from an offset point to estimate the differential boundary condition. 52
- 6.3 (*Left*) Estimating the primal normal derivative: the pointwise WoS gradient estimator evaluated at an offset point, versus the backward-difference stencil of Equation (6.11), which differences the prescribed boundary value against a primal estimate at the offset. (*Right*) Estimating the product $u\dot{u}$: reusing the same walk for both factors correlates the estimates and biases the product, while using each walk for only one factor is unbiased but wasteful; the U-statistic averages all cross-walk pairs, remaining unbiased while reusing every walk. . . 56
- 6.4 Shape from diffusion: starting from a sphere, boundary geometry evolves to match solution measurements on a surrounding box. Because screened diffusion low-passes the boundary signal, high-frequency surface detail leaves no measurable imprint and is not recovered (bottom row). 57
- 6.5 Optimization-driven thermal design: the heating elements of a toaster (Dirichlet boundary) are optimized so that the induced temperature field matches a target profile on a slice of bread (Neumann boundary). For partial updates to the target, derivatives are computed only over the modified region. 58

- 6.6 *(Left)* Noisy derivatives suit stochastic optimization: at equal time, cheap noisy estimates make more progress toward the target, while at equal iterations refined estimates land more precisely—inviting a schedule that increases the sample count as the iterate approaches a solution. *(Right)* Finite differences require a separate primal solve per parameter, whereas a single collection of nested walks estimates derivatives for all parameters simultaneously: at equal time the derivative noise is lower, and the gap widens with parameter count. 58
- 6.7 Optimizing geometric representations that conventional solvers cannot handle directly. *(Left)* Inverse diffusion curves: the control points, tangent handles, and RGB boundary data of Bézier curves are optimized jointly to match a target image. *(Right)* Shape inflation with an implicit boundary: the positions and scales of harmonic radial basis functions are optimized to match a target heightfield, with topological changes coming for free. 59
- 7.1 A heat lamp radiatively warms a pizza while conduction redistributes heat across its surface, modeled as a coupled system of integral equations—radiative transfer evaluated by ray tracing, conduction by walk on spheres *(left)*. Effective sampling strategies for the coupled system interleave operations from both algorithms, which ordinarily requires merging two solvers into one monolithic estimator program; expressed as rewrites of the composed integral program, the same strategies take a few lines, and remain robust as heating profiles concentrate *(right)*. Figure reproduced from Mu et al. [2026], work currently under submission. 61
- 7.2 *(Left)* Stochastic opaque solids extend solid geometry with uncertainty, much as microparticle models capture geometric complexity, replacing deterministic queries with distributional equivalents. *(Right)* Probabilistic free-flight queries on uncertain solids improve neural surface reconstruction over more heuristic volumetric models [Miller et al. 2024]; a closest-point analogue of the same query would unlock walk on spheres on these representations. 63
- 7.3 The three ways PDEs and neural fields already meet, illustrated left to right. *(Left)* Solving a PDE on a neural field, conductive thermal Gaussian surfels reconstruct both the surface and interior temperature of an object from multi-view thermal images [Liu et al. 2026]. *(Center)* Representing a neural field with PDE machinery, a regularized dipole sum turns a structure-from-motion point cloud into a point-based implicit surface and radiance field optimized by differentiable rendering [Chen et al. 2024a]. *(Right)* Processing a neural field through a PDE solve, Sobolev preconditioning recovers a clean surface from an outlier-ridden point cloud, where the unpreconditioned baseline and an alternative stochastic preconditioning either fail or oversmooth [Su et al. 2026]. 64

Chapter 1

Introduction

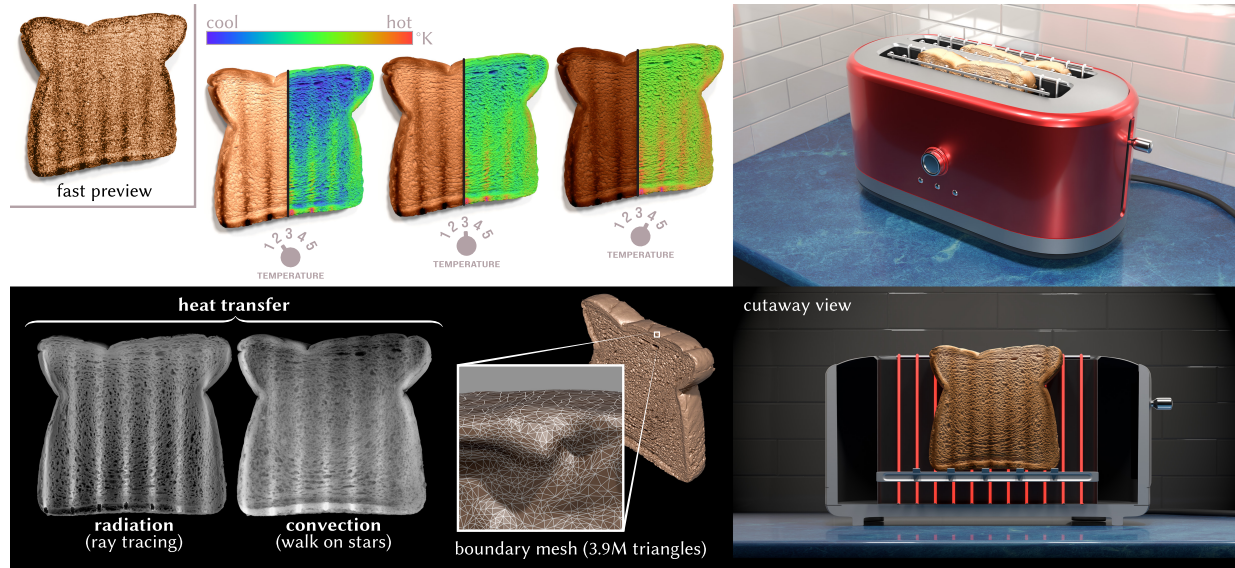


Figure 1.1: The walk on stars (WoSt) family of estimators developed in this thesis enables a richer class of problems than the original walk on spheres (WoS) method. Here, for instance, we simulate diffusive convective heat transfer from a toaster (Dirichlet) to a piece of bread (Neumann) by solving a Laplace equation with mixed boundary conditions (top and bottom right), complementing the radiative transfer computed via ray tracing (bottom left). Just like ray tracing, we can simulate directly on the full high-resolution data (bottom center) without generating a volume mesh or forming a global stiffness matrix. Since results are progressive, we can preview how the toast will look faster than it takes to toast a real piece of bread (top left).

1.1 Motivation

Recent work develops *walk on spheres* as a Monte Carlo alternative to finite-element methods for linear elliptic PDEs [Sawhney and Crane 2020]. The aim is not to replace them wholesale, but to offer an attractive option precisely where global discretization struggles: on complex, detailed, or otherwise challenging geometry. Walk on spheres achieves this aim for the same

reason Monte Carlo rendering does: both recast a globally coupled problem—an elliptic PDE, or light transport—as a recursive integral equation, then estimate it by tracing local random walks instead of discretizing the domain. Requiring only pointwise access to the geometry through local queries, much as ray tracing queries a scene only along individual rays, the method inherits many of the same algorithmic advantages that make Monte Carlo rendering attractive: robustness to geometric complexity, output-sensitive evaluation, flexibility to geometric representation, and trivial parallelism. Yet walk on spheres still lacks many of the core capabilities that have made rendering a ubiquitous tool across graphics, vision, and imaging. Across each of these capabilities, the contrast with rendering is stark:

- it is largely confined to Dirichlet boundary conditions, where rendering algorithms draw on a broad set of physically accurate reflectance models [Walter et al. 2007; Heitz et al. 2016; Yan et al. 2015, 2016] (Figure 1.2);
- it has no analogue of participating media, the statistically described volumetric matter—clouds, smoke, suspensions—that rendering simulates routinely [Novák et al. 2018];
- it has limited variance reduction and reuse machinery, where rendering has an arsenal of techniques developed over decades [Ward et al. 1988; Veach and Guibas 1995; Vorba et al. 2014; Bitterli et al. 2020]; and
- where rendering has a mature set of differentiable variants that enable inverse and optimization tasks [Li et al. 2018; Zhang et al. 2020; Nimier-David et al. 2020], walk on spheres has none.

This thesis argues that such capabilities are essential to making walk on spheres a practical solver, and that the structural similarities sketched above let us not merely take inspiration from rendering but often adapt its machinery and borrow its insights directly.

Rendering, too, was not an overnight success. It began as a geometrically capable estimator that lacked nearly everything else: the physically accurate materials, the volumetric models, the variance reduction and reuse, the differentiable formulations all came later, developed over three decades into the framework rendering is today. Even its proponents were once measured about its prospects,

“In summary, pure MCPT has only two advantages—it is so dumb that it doesn’t get hit by big scenes, and it is easy to implement.”

—Peter Shirley [Shirley 1996].

What drove the field to build all of this machinery was precisely the appeal of the geometric core: an estimator robust to complexity, indifferent to representation, and trivially parallel was worth the considerable effort of building everything else on top of it. Walk on spheres offers the same core, and we argue the same effort is warranted—with the advantage that much of rendering’s machinery can be adapted rather than reinvented, often drawing on the same underlying mathematics. We explore these connections throughout the thesis.

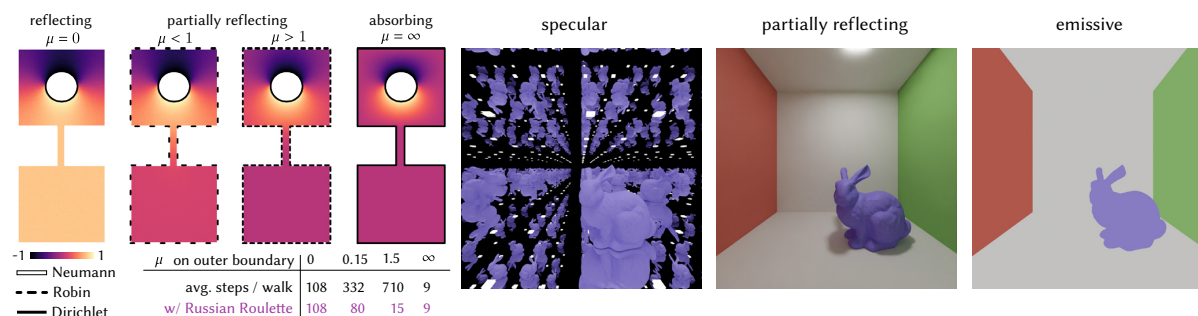


Figure 1.2: (Left) A non-negative Robin coefficient μ interpolates between reflecting Neumann ($\mu = 0$) and absorbing Dirichlet ($\mu \rightarrow \infty$) conditions, which prescribe boundary derivatives and values, respectively. As μ increases and the boundary becomes less reflecting, Russian roulette probabilistically terminates walks on the Robin boundary, reducing the average number of steps per walk. (Right) The rendering analogue: a room with perfectly specular surfaces reflects all incident radiance, like a Neumann boundary; realistic scenes include materials that only partially reflect; and purely emissive surfaces behave like prescribed Dirichlet values. As in rendering, richer material models yield more useful and realistic solutions.

It is also worth asking why these developments make sense to pursue now. The core algorithms underlying both rendering and walk on spheres have been around since the advent of computing, dating to the 1950s [Kahn 1950; Muller 1956]. So why have they only recently been pursued in earnest? Rendering is a good place to start looking for an answer to this question. For most of that history, the geometry one could actually put into a computer was simple enough for mesh-based methods to handle; it was only in the 1990s, with visual effects, computer-animated film, 3D games, and architectural visualization, that rendering encountered geometry complex enough to motivate grid-free Monte Carlo as a serious tool. Walk on spheres now finds itself at a similar moment. The geometry we wish to analyze, process, or optimize is increasingly large, irregular, and difficult to discretize robustly—whether measured geometry such as the point clouds and reconstructions produced by 3D scanning, the detailed models output by modern design and generative tools, or the intricate engineered structures that additive manufacturing has made practical to fabricate. It is also defined through an increasing variety of representations—neural implicit surfaces, signed-distance fields, and point clouds among them—for which generating a conforming volumetric mesh is especially difficult. This is the kind of setting where a solver built on local geometric queries becomes appealing.

A final force accelerates the trend in both rendering and walk on spheres: as parallel hardware grows ever more powerful, the cost of sampling falls, lowering the geometric complexity at which a grid-free approach pays off. Just as this shift has carried Monte Carlo rendering from offline computation into interactive and real-time applications, it stands to widen the reach of walk on spheres as well, with GPU implementations already emerging [Sawhney 2026].

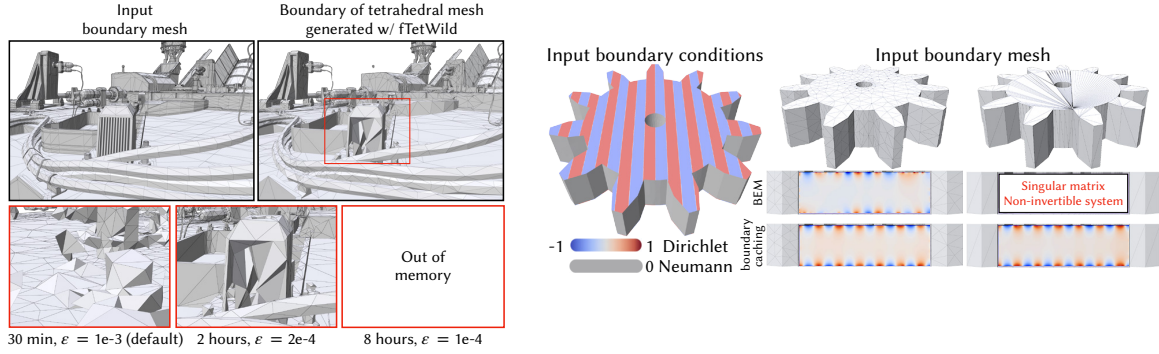


Figure 1.3: (Left) Generating tetrahedral meshes for accurate FEM simulation is challenging on detailed models: state-of-the-art tools either fail to capture important detail or run out of memory at finer tolerances—here, fTetWild [Hu et al. 2020] on the Mars rover of Figure 3.1 with 64 GB of memory, while TetGen [Si 2015] cannot tetrahedralize the model at all due to self-intersections. (Right) Even for simple boundary conditions, boundary element methods suffer large global errors without significant mesh refinement, due to local aliasing of boundary data, and can fail outright on irregular elements (top row).

1.2 Approach and Scope

The thesis takes an integral-first approach. Each new capability begins by identifying a classical integral equation appropriate to the problem at hand—Green’s third identity for general first-order linear boundary conditions, an expected mean value relation for participating media, the boundary integral equation for global solution caching, and a nested boundary value problem for derivatives—then performing a small number of principled modifications, and finally forming a Monte Carlo estimator over the resulting integrals. The estimators produced this way are not isolated tricks but a coherent computational framework that mirrors the organization of rendering itself: surface and volume variants share machinery; a caching scheme reduces to plugging an existing solver pipeline into a final-gather-like reconstruction step [Jensen 1996]; and—as in differentiable rendering—derivatives reduce to a primal-like algorithm augmented with a small amount of additional sampling. To keep these constructions readable, we prefer the simplest PDEs and conditions that still expose the ideas—most often the homogeneous Laplace equation—as source terms and the screened Poisson extension are handled by standard means already in the literature [Sawhney and Crane 2020; Sawhney et al. 2022]. What emerges across these chapters is more than a set of estimators: it is a working dictionary between rendering and walk on spheres that subsequent work has continued to extend [Ho and Owen 2026b; Huang et al. 2025; Hsu et al. 2026].

We do not pit walk on spheres against finite-element or boundary-element methods. These are mature, capable frameworks (Figure 1.3), and prior work [Sawhney and Crane 2020] has already established that walk on spheres offers an orthogonal set of capabilities whose relative advantage is strongly application- and regime-dependent—favoring grid-free methods on complex geometry with sparse evaluation, favoring mesh-based methods on geometry of modest complexity with dense solves, and shifting continually as new acceleration techniques are developed on each side [Chen et al. 2024b, 2025]. Our aim is not a universal speedup but a broader set

of problems within which walk on spheres can be considered at all. Each capability is accordingly compared against the closest available Monte Carlo approach: existing reflectance-style schemes for first-order conditions [Mascagni and Simonov 2004; Maire and Tanré 2012; Zhou et al. 2015], ensemble averaging and homogenization for participating media [Marchenko and Khruslov 2008; Papanicolaou 1995; Giunti et al. 2018], and—where no prior method exists—against the bar of demonstrating that the capability can be built without sacrificing convergence. The space of PDEs, domains, and conditions is too broad for any single benchmark to be decisive in the way a benchmark scene is for rendering, so we emphasize categorical capabilities over headline error numbers.

The thesis is deliberately bounded in two ways. First, it stays within second-order linear elliptic PDEs—Laplace and screened Poisson, under various boundary conditions—and does not address vector-valued or higher-order linear systems such as Stokes flow or linear elasticity, nor indefinite problems such as Helmholtz, nor parabolic and transient problems. We expect the patterns developed here to transfer: recent work already uses walk on spheres as a subroutine in tackling some of these settings [Rioux-Lavoie et al. 2022; Bati et al. 2023; Jain et al. 2024], and the underlying ideas around convergence, sampling, acceleration, differentiability, and stochastic geometry should carry over. Second, the variance reduction and reuse we develop is of the broad, general-purpose kind—caching schemes that apply across an entire problem—rather than the tailored, capability-specific refinements that would follow within any one estimator. This mirrors rendering’s own development: the difficult step has always been the first converging estimator for a new setting—participating media, differentiable variants, and so on—after which decades of tailored refinement followed. Given the breadth of the PDE landscape and the strong dependence of useful optimizations on the targeted application, this seems to us the right order of operations.

1.3 Contributions

This thesis contributes four Monte Carlo estimators, one for each of the gaps relative to rendering identified at the outset—boundary conditions, participating media, dense reuse, and differentiable variants—together with the conceptual machinery that connects them to their rendering counterparts.

- **Walk on stars.** A recursive boundary integral equation, derived via Green’s third identity on star-shaped subdomains, that handles all first-order linear boundary conditions—Dirichlet, Neumann, and Robin—in a single walk-on-spheres-style estimator. Convergence on nonconvex geometry follows from an operator-theoretic spectral radius bound—a guarantee earlier boundary-walk methods lacked. The estimator is a direct analogue of the rendering equation: a reflectance factor plays the role of an albedo, a star-shaped subdomain the role of a visible region, and the Poisson kernel the role of a geometry term.
- **Volumetric walk on spheres.** An extension of walk on spheres to random participating media, formulated as a Monte Carlo estimator for the expected mean value relation taken over configurations of a Boolean particle model. The walk replaces deterministic closest-point queries with conditional closest-particle sampling—the PDE counterpart of free-flight sampling in volume rendering—with a memory mechanism that keeps successive samples

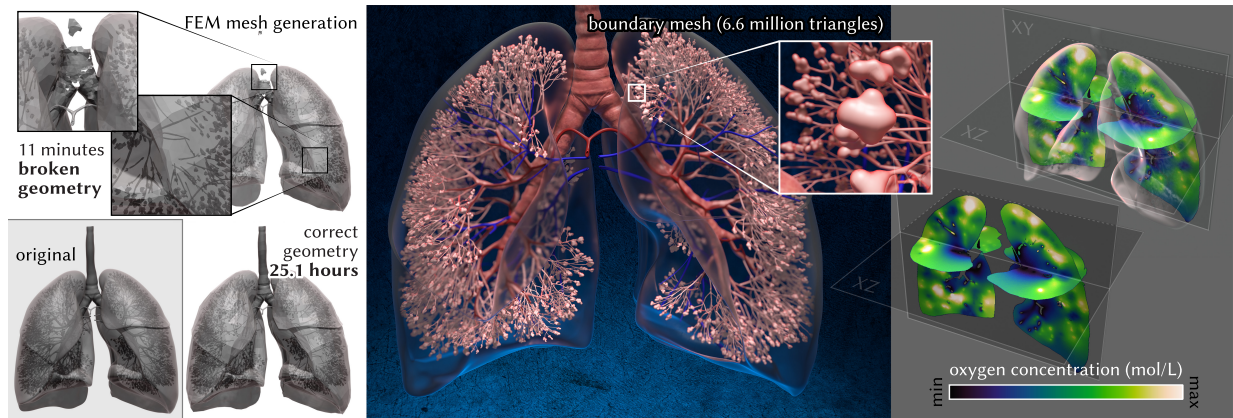


Figure 1.4: Where will oxygen flow at the beginning of a breath? Walk on stars simulates gas exchange in a lung model whose thin features defeat volumetric meshing—the kind of problem the contributions of this thesis make accessible: general boundary conditions, evaluated directly on complex geometry (Chapter 3).

consistent with prior ones. A single walk estimates the mean solution without ever realizing the full random geometry, and the same memory mechanism enables correlated coupling with a volume renderer. For reflecting particles, closest-silhouette queries give the analogous volumetric walk on stars.

- **Boundary value caching.** A reuse scheme that decouples dense interior evaluation from boundary sampling through the boundary integral equation: cached estimates on the boundary reconstruct the solution at every interior point without a per-point walk, and because the underlying kernels are harmonic in the evaluation point, the reconstruction satisfies the governing PDE by construction rather than only in expectation. Storage scales with surface area instead of volume, and the absence of an occlusion term in the boundary integral makes reuse globally unrestricted—unlike rendering’s virtual point lights, which see only what they can illuminate. Practical considerations—singular kernels, fast summation, and localized region-of-interest reuse—complete the scheme.
- **Differential walk on spheres.** A demonstration that the derivative of the walk-on-spheres solution with respect to a geometric parameter satisfies its own boundary value problem—homogeneous, with boundary data depending on the primal solution’s normal derivative—which can be solved by a nested walk that reuses the same geometric query machinery as the primal solver. The resulting differentiable PDE solver requires no automatic differentiation through the stochastic algorithm, remains agnostic to boundary representation, and produces stochastic gradients—unbiased up to the standard ε -shell approximation—naturally suited to stochastic optimization.

Taken together, these four constructions are early steps in a longer project: bringing into Monte Carlo PDE solving the breadth of capability that rendering has accumulated over three decades.

Chapter 2

Background

The estimators developed in this thesis rest on two ideas: Monte Carlo estimation of integrals, and the representation of PDE solutions as recursive integral equations. This chapter reviews both at the level of generality needed by later chapters, which introduce further background where it is first used.

2.1 Monte Carlo Estimation

Consider an integral over a domain $\Omega \subset \mathbb{R}^d$,

$$I = \int_{\Omega} f(x) \, dx. \quad (2.1)$$

A *Monte Carlo estimator* replaces exact integration with an average over random samples: drawing $x_1, \dots, x_N$ independently from a probability density p on Ω ,

$$\hat{I}_N := \frac{1}{N} \sum_{i=1}^N \frac{f(x_i)}{p(x_i)}. \quad (2.2)$$

Whenever p is positive wherever f is nonzero, the estimator is *unbiased*, $\mathbb{E}[\hat{I}_N] = I$, and its error is characterized entirely by its variance. Writing $\sigma^2 := \text{Var}[f(x)/p(x)]$ for the single-sample variance, independence gives $\text{Var}[\hat{I}_N] = \sigma^2/N$, so when σ is finite the root-mean-square error decays as

$$\sqrt{\mathbb{E}\left[\left(\hat{I}_N - I\right)^2\right]} = \frac{\sigma}{\sqrt{N}} = O\left(N^{-1/2}\right). \quad (2.3)$$

The defining feature of this convergence rate is what it does *not* depend on: the dimension d of the domain or the smoothness of f . A tensor-product quadrature rule achieves only $O\left(N^{-k/d}\right)$ for integrands with k derivatives, whereas Monte Carlo proceeds at the same rate in every dimension [Owen 2013]. This dimension independence, together with the ability to evaluate $\hat{I}_N$ pointwise and in parallel, is what the PDE estimators of this thesis inherit.

The constant σ , by contrast, is entirely in our control through the choice of density. If $p \propto f$ (for nonnegative f), every sample returns the constant I and the variance vanishes; more generally, the closer p matches the shape of f , the smaller σ becomes. This principle, *importance sampling*, drives nearly every algorithmic decision in the chapters ahead, where densities are chosen to cancel singular kernels exactly.

2.2 Volterra Integral Equations

Monte Carlo estimation extends from definite integrals to integral *equations*, where the unknown is a function defined implicitly in terms of integrals of itself. A linear integral equation of the second kind has the form

$$u(x) = \int_{A(x)} k(x, y) u(y) dy + b(x), \quad (2.4)$$

where k is a known kernel, b a known source term, and $A(x) \subseteq \Omega$ the integration domain. When A is fixed—the same for every x —the equation is *Fredholm*; the rendering equation, integrating over a fixed scene boundary, is the archetype. When $A(x)$ varies with the evaluation point, the equation is *Volterra* [Kalos and Whitlock 2008]. The recursive representations of elliptic boundary value problems derived in this thesis are of the latter type: each accounts for the PDE on a local subdomain around x , and the subdomain shifts at every step.

Writing C for the integral operator in (2.4), the equation reads $u = Cu + b$, and repeatedly substituting the equation into itself produces the *Neumann series*

$$u = \sum_{i=0}^{\infty} C^i b, \quad (2.5)$$

which converges to the unique solution when some iterate of the operator is a contraction, $\|C^i\| < 1$ [Kalos and Whitlock 2008; Arvo 1995]. The series suggests the estimator: a *single-sample recursive estimator* estimates the integral in (2.4) with one sample $y \sim p(\cdot | x)$ and recurses,

$$\hat{u}(x) = \frac{k(x, y)}{p(y | x)} \hat{u}(y) + b(x). \quad (2.6)$$

Crucially, we draw a *single* sample at each recursive step. Drawing several would branch the computation into a tree that grows exponentially with depth. One sample keeps the work linear in the walk length, and we reduce variance by averaging many independent walks. Unrolling the recursion yields a random walk $x_0, x_1, x_2, \dots$ that accumulates the source $b(x_k)$ at each step, weighted by a *throughput*—the running product of the per-step factors k/p . The walk is a stochastic evaluation of the Neumann series, with one path standing in for the entire sum. Because the series is infinite, the walk needs a termination rule: truncating at a fixed depth introduces bias, while *Russian roulette*—continuing with some probability q at each step and reweighting survivors by $1/q$ —terminates paths at finite expected length without bias. For the estimator to converge with finite variance, the throughput must contract along a path, mirroring the operator condition $\|C^i\| < 1$; Section 3.3 makes this correspondence precise. Monte Carlo

rendering is the proof of concept: path tracing [Kajiya 1986] is exactly the estimator (2.6) applied to the Fredholm rendering equation, and the chapters ahead apply the same machinery to Volterra representations of elliptic boundary value problems (Chapter 3). The next section derives the simplest such representation.

2.3 PDEs as Integral Equations

Consider the Laplace equation with Dirichlet boundary conditions and no source term,

$$\begin{aligned}\Delta u(x) &= 0, & x \in \Omega, \\ u(x) &= g(x), & x \in \partial\Omega,\end{aligned}\tag{2.7}$$

where the *Laplacian* $\Delta u := \sum_{i=1}^d \partial^2 u / \partial x_i^2$ measures the deviation of u at x from its average over neighboring points. Setting it to zero therefore smoothly interpolates the boundary data g into the interior of the domain—an intuition the mean value property will shortly make exact.

Green’s Representation. To express (2.7) as an integral equation, we introduce two kernels that recur throughout this thesis: the *Green’s function* G , satisfying $\Delta G(x, \cdot) = \delta_x$ on a subdomain $A(x) \subseteq \Omega$, and the associated *Poisson kernel* $P := \partial G / \partial \vec{n}_y$ [Evans 1998; Duffy 2015]. We write the subdomain as a function of position because we are free to select it separately for every evaluation point—the freedom that drives everything that follows. Green’s third identity represents the solution as an integral over the subdomain boundary [Costabel 1987],

$$\alpha(x) u(x) = \int_{\partial A(x)} P(x, y) u(y) - G(x, y) \frac{\partial u(y)}{\partial \vec{n}_y} dy, \quad \alpha(x) = \begin{cases} 1 & x \in A(x) \\ \frac{1}{2} & x \in \partial A(x) \\ 0 & \text{otherwise,} \end{cases}\tag{2.8}$$

where the factor α accounts for the location of the evaluation point: on a smooth boundary only half of the kernel’s singularity lies within the subdomain. In this section every evaluation point is interior, so $\alpha = 1$ throughout; evaluation on the boundary itself becomes essential in Chapter 3.

The Mean Value Property. The representation (2.8) holds for *any* choice of subdomain and any boundary conditions imposed on G —freedom we now exploit to eliminate the unknown normal derivative. Take $A(x)$ to be the largest empty ball $B(x, d(x))$, where $d(x)$ is the distance from x to $\partial\Omega$, and let G be the Green’s function of that ball, which vanishes on its surface. The normal derivative term in (2.8) disappears, and the kernels take simple closed forms via the method of images [Duffy 2015]: evaluated from the center of a ball of radius r in 3D,

$$G(x, y) = \frac{1}{4\pi} \left(\frac{1}{\|x - y\|} - \frac{1}{r} \right), \quad P(x, y) = \frac{\langle \vec{n}_y, y - x \rangle}{4\pi \|x - y\|^3}.\tag{2.9}$$

On the sphere itself, $\|x - y\| = r$ and $\langle \vec{n}_y, y - x \rangle = r$, so the Poisson kernel is constant, $P = 1/4\pi r^2$, and (2.8) collapses to the *mean value property* of harmonic functions [Evans 1998]: $u(x)$

equals the average of u over any empty sphere centered at x ,

$$u(x) = \frac{1}{|\partial B(x, r)|} \int_{\partial B(x, r)} u(y) \, dy. \quad (2.10)$$

Walk on Spheres. Equation (2.10) is precisely a Volterra integral equation of the kind described in Section 2.2: the integration domain $\partial B(x, d(x))$ moves with the evaluation point. Sampling y uniformly on the sphere importance samples the constant kernel perfectly, so the single-sample recursive estimator (2.6) has throughput identically one. What the recursion lacks is a source term: the walk bounces from sphere to sphere forever, approaching $\partial\Omega$ but never reaching it. The standard remedy introduces an ε -shell, the set of interior points within ε of the boundary,

$$\partial\Omega_\varepsilon := \{x \in \Omega : d(x) < \varepsilon\}. \quad (2.11)$$

Whenever the walk enters $\partial\Omega_\varepsilon$, it terminates and reads the boundary data at the closest boundary point $\bar{x}$,

$$\hat{u}(x_k) = \begin{cases} g(\bar{x}_k), & x_k \in \partial\Omega_\varepsilon, \\ \hat{u}(x_{k+1}), & x_{k+1} \sim \mathcal{U}(\partial B(x_k, d(x_k))). \end{cases} \quad (2.12)$$

The termination plays the role of the source b in (2.4), and makes convergence work: walks take $O(\log 1/\varepsilon)$ expected steps to terminate, at the cost of a small bias controlled by ε [Binder and Braverman 2012]. Estimator (2.12) is the *walk on spheres* (WoS) algorithm of Muller [1956], revived for geometry processing by Sawhney and Crane [2020]. Notice that the estimator never needed Russian roulette: a Dirichlet boundary *absorbs* every walk that wanders near it, so the geometry itself guarantees termination. Boundary conditions that instead reflect walks back into the domain provide no such guarantee, and Chapter 3 will restore termination probabilistically, with Russian roulette driven by a reflectance.

Screened Operators. The same construction extends to the screened operator $\Delta - \sigma$. The method of images again yields closed-form kernels for the ball [Sawhney et al. 2023, Appendix A]: in 3D,

$$G^\sigma(x, y) = \frac{1}{4\pi} \frac{\sinh((r - \|x - y\|)\sqrt{\sigma})}{\|x - y\| \sinh(r\sqrt{\sigma})}, \quad P^\sigma(x, y) = Q^\sigma(x, y) P(x, y), \quad (2.13)$$

where P is the Poisson kernel of (2.9) and $Q^\sigma \in [0, 1)$ is a closed-form attenuation factor [Sawhney et al. 2023, Equation 42]. Screening thus damps the Poisson kernel: each step of a walk loses a fraction of its contribution to absorption, exactly as a participating medium attenuates light. Both kernels reduce to (2.9) as $\sigma \rightarrow 0$.

2.4 The Walk on Boundary Alternative

Before building our own estimator, we pause to contrast walk on spheres with its principal grid-free alternative, whose failure modes motivate several of the design choices in the chapters

that follow. Walk on spheres is not the only way to turn a boundary value problem into a recursive Monte Carlo estimator. A prominent alternative, walk on boundary (WoB), recasts the problem as an integral equation on the boundary itself. Where WoS walks through the interior until it nears the boundary, WoB represents the solution as a boundary integral and walks along the boundary, stepping from one surface point to the next. It originates in numerical analysis [Sabelfeld and Simonov 1994] and was introduced to graphics by Sugimoto et al. [2023]. For the Dirichlet Laplace problem, a double-layer representation with unknown density v gives the integral equation

$$v(x) = \int_{\partial\Omega} 2 \frac{\partial G}{\partial \vec{n}_y}(x, y) v(y) dy + 2g(x), \quad (2.14)$$

which the same single-sample recursion as the rendering equation solves, with the dipole kernel $2 \partial G / \partial \vec{n}$ as a scattering term and $2g$ as emission. Walk on boundary is appealing because the kernel is the solid angle the boundary subtends, so ray casting importance samples it and WoB reuses a renderer's ray queries directly. It also handles exterior problems and needs no ε -shell, since every vertex already lies on the boundary.

This convenience gives up the guarantee that makes WoS reliable. Because the integration domain is the fixed boundary rather than a shrinking ball, (2.14) is *Fredholm* rather than Volterra, and its kernel is nonzero everywhere [Sugimoto et al. 2023]. In operator form $v = Hv + 2g$, the solution is the Neumann series

$$v = \sum_{i=0}^{\infty} H^i 2g, \quad (2.15)$$

but the operator H does not contract. Its kernel integrates to one, the analogue of a perfectly reflecting scene, so the terms $H^i 2g$ never decay. The series does not converge in the ordinary sense, and truncating it at a fixed length is biased. WoB sidesteps this with a *modified Neumann series*, a transformation that does converge and that the original authors interpret as an analytic continuation [Sugimoto et al. 2023]. It averages consecutive partial sums, which cancel by alternating sign, and amounts in practice to halving the last vertex's contribution. Russian roulette, the usual route to an unbiased termination, does not directly apply. The modified series depends on weighting consecutive terms together, and the independent random termination of Russian roulette breaks that pairing, so an unbiased scheme is left to future work. A second difficulty is variance. Even for Dirichlet problems the double-layer kernel changes sign with the orientation of the boundary it meets, so on nonconvex geometry a ray strikes the boundary at several points whose contributions alternate in sign and branch the walk. Signed integrands of this kind resist importance sampling by a positive density [Owen 2013], and taming the variance likely calls for a positivization into single-signed parts, as differentiable rendering does for signed BRDF derivatives [Belhe et al. 2024].

Mixed boundaries are harder still. A single-layer representation with density ϕ gives, on the Dirichlet part, a first-kind equation

$$g(x) = \int_{\partial\Omega} G(x, y) \phi(y) dy, \quad (2.16)$$

in which the unknown sits only inside the integral, leaving nothing to recurse on. WoB makes it estimable by multiplying through by a constant k and adding ϕ to both sides,

$$\phi(x) = \phi(x) - k \int_{\partial\Omega} G(x, y) \phi(y) dy + k g(x), \quad (2.17)$$

a Richardson iteration whose step size k is a free numerical parameter, not part of the problem itself. The solution does not depend on k , but its convergence does. The iteration converges only over a range of k and fails when k is too large, so k is set by trial and error. Choosing such a step or a preconditioner would classically draw on a spectral analysis of the operator [Saad 2003], but a single-sample walk has no access to that spectrum. Later work has extended WoB to more conditions and equations, including a thesis devoted largely to it [Sugimoto 2025]. Whether a robust, problem-independent contraction exists for WoB is open.

Yet in the cases where it has been applied, such as a recent Monte Carlo fluid solver [Sugimoto et al. 2024], the domains tend to be relatively simple and the bias tolerable. It gives up the geometric scalability to complex domains that motivates grid-free Monte Carlo in the first place. On nonconvex geometry its variance grows without bound, as we demonstrate empirically in Chapter 3. By contrast, WoS retains this scalability. It breaks the global solve into local subproblems, each a mean value relation on an empty ball, so convergence is geometric rather than spectral. A walk reaches the boundary in a finite expected number of steps on any domain [Binder and Braverman 2012], with no operator to precondition and no parameter to tune. Its one approximation is the ε -shell that terminates the walk, whose bias is minimal and interpretable [Sawhney and Crane 2020]. In Section 3.3 we discuss contraction as it relates to walk on stars, the extension of WoS to general first-order linear boundary conditions, and show that in those cases it too retains the contractive properties that make both WoS and rendering algorithms practical and robust.

2.5 Poisson Boolean Model

Monte Carlo rendering and walk on spheres algorithms interact with geometry exclusively through local queries—ray intersections, closest-point distances, uniform sphere sampling, and silhouette queries. All assume the geometry can be queried *deterministically*. Media composed of microscopic particles break this assumption: a cloud of water droplets or an electrolyte of ions comprises far too many elements to store, let alone query. The remedy is to describe the geometry *statistically* and answer queries by sampling. We focus on the standard such model for independent particles, the *Poisson Boolean model* (PBM); our presentation follows Last and Penrose [2018, Chapters 16–17] and Chiu et al. [2013, Chapters 2–3].

The Model. A stochastic microparticle geometry $\mathcal{O} \subset \mathbb{R}^3$ follows the PBM with density $\lambda : V \rightarrow \mathbb{R}_{\geq 0}$, defined over a bounded region $V \subset \mathbb{R}^3$ containing the domain, and particle size R if it is a union of balls whose centers form a Poisson point process,

$$\mathcal{O} := \bigcup_{n=1}^N B(c_n, R), \quad N \sim \text{Poisson}\left(\int_V \lambda(x) dx\right), \quad p(c_n | N) \propto \lambda(c_n), \quad (2.18)$$

that is, the number of particles is Poisson-distributed, and conditionally on their count the centers are independent with density proportional to λ . When λ is constant we call the model *homogeneous*. The PBM never materializes the union in (2.18); instead, every geometric query is answered in distribution, by sampling only the particles the query would touch.

Sampling Along Rays. For a ray $r_{x,\vec{\omega}}(t) := x + t\vec{\omega}$, the distance to the first particle intersection follows the *free-flight distribution* [Last and Penrose 2018],

$$p_{x,\vec{\omega}}^{\text{ff}}(t) = \sigma_t(r_{x,\vec{\omega}}(t)) \exp\left(-\int_0^t \sigma_t(r_{x,\vec{\omega}}(s)) \, ds\right), \quad \sigma_t := \lambda \pi R^2, \quad (2.19)$$

an exponential distribution whose rate is the *extinction coefficient* σ_t , the product of particle density and cross-sectional area. This is precisely the transmittance law of exponential participating media: the PBM is the microparticle geometry underlying classical volume rendering [Novák et al. 2018].

Sampling Closest Points. WoS instead queries the closest point on the geometry, which—since all particles share the radius R —lies on the particle whose center is nearest to x . The distance from x to that nearest center follows a law structurally identical to the free-flight distribution (2.19), with growing spheres in place of an advancing ray [Last and Penrose 2018, Section 7.4]:

$$p_x^{\text{dc}}(t) = \left(\int_{\partial B(x,t)} \lambda(y) \, dy\right) \exp\left(-\int_{B(x,t)} \lambda(y) \, dy\right), \quad (2.20)$$

the density of encountering the first center at distance t : the rate of centers on the sphere’s surface, attenuated by the probability that the ball it encloses is empty—just as (2.19) attenuates the extinction at the ray tip by the transmittance of the segment behind it. In the homogeneous case, (2.20) says the *cubed* distance to the nearest center $c(x)$ is exponentially distributed,

$$\|x - c(x)\|^3 \sim \text{Expo}\left(\frac{4}{3}\pi\lambda\right), \quad (2.21)$$

so sampling the nearest center costs one exponential variate and one uniform direction: draw $\xi \sim \text{Expo}\left(\frac{4}{3}\pi\lambda\right)$ and place $c(x)$ uniformly on the sphere $\partial B\left(x, \xi^{1/3}\right)$ [Chiu et al. 2013, Section 2.5]. Since particles are balls of radius R , the closest point on $\partial\mathcal{O}$ lies along the segment toward that center,

$$q_{\partial\mathcal{O}}^*(x) = x + (\|x - c(x)\| - R) \frac{c(x) - x}{\|c(x) - x\|}, \quad (2.22)$$

with a negative distance $\|x - c(x)\| - R < 0$ indicating that x lies inside a particle. The closest-point distribution this procedure samples is known analytically [Last and Penrose 2018, Section 16.3].

Thinning. When the density λ varies spatially, both samplers generalize by *thinning* [Lewis and Shedler 1979]: choose a majorant $\bar{\lambda} \geq \lambda$, conceptually fill the domain with fictitious density until it is homogeneous at rate $\bar{\lambda}$, sample candidate centers from the homogenized model in

order of increasing distance—accumulating cubed-distance increments $\xi_s \sim \text{Expo}(\frac{4}{3}\pi\bar{\lambda})$ —and accept each candidate with probability $\lambda(c_s)/\bar{\lambda}$. The first accepted candidate is the closest center. Readers familiar with volume rendering will recognize this as the closest-point analogue of delta tracking, which performs the same homogenize-and-reject procedure along rays to sample free-flight distances in heterogeneous media [Raab et al. 2006; Novák et al. 2018]. Chapter 4 builds its estimators on exactly this sampler, generalized to condition on the history of a walk.

2.6 A Tour of Monte Carlo Rendering

The machinery of this chapter—recursive estimators for integral equations, evaluated over sampled geometry—is not hypothetical: Monte Carlo rendering has spent four decades proving that it scales. This section is a brief tour of that success, one equation per stop, each foreshadowing a chapter of this thesis.

Ballistic Transport Equation. Photorealistic rendering asks a deceptively simple question—given a scene’s geometry, materials, and lights, what does the camera see? The image it records is the solution of a PDE. Light transport is governed by a first-order linear equation: in free space radiance is constant along rays, and at surfaces it is reborn by emission and scattering [Arvo 1995],

$$\vec{\omega} \cdot \nabla L(x, \vec{\omega}) = 0, \quad x \in \Omega, \quad L(x, \vec{\omega}) = L_e(x, \vec{\omega}) + L_s(x, \vec{\omega}), \quad x \in \partial\Omega, \quad \langle \vec{n}, \vec{\omega} \rangle > 0. \quad (2.23)$$

Like the boundary value problems of this thesis, this is a PDE whose solution is coupled globally through the geometry—every surface point illuminates every other—and the history of rendering is the history of resolving that coupling stochastically.

The Rendering Equation and Path Tracing. Path tracing is what made physically based rendering practical at production scale. Films today are rendered on scenes of billions of primitives, with no meshing of the space between surfaces and no global system of equations to solve. That capability rests on a single recursive integral equation. The in-scattered radiance L_s couples outgoing to incident light through the bidirectional scattering distribution function (BSDF) f , turning the boundary condition of (2.23) into the *rendering equation* [Kajiya 1986],

$$L(x, \vec{\omega}) = L_e(x, \vec{\omega}) + \int_{S_+^2} f(x, \vec{\omega}, \vec{\omega}') \langle \vec{n}, \vec{\omega}' \rangle L(r_{x, \vec{\omega}'}^*, -\vec{\omega}') d\vec{\omega}', \quad (2.24)$$

where $r_{x, \vec{\omega}'}^*$ is the first surface hit by the ray from x in direction $\vec{\omega}'$. This is a Fredholm equation of the second kind, and *path tracing* is exactly the single-sample recursive estimator (2.6) applied to it: sample a direction, cast a ray, recurse, and terminate by Russian roulette with the surface albedo as survival probability. Chapter 3 constructs the same estimator for the Volterra equations of elliptic boundary value problems—with the Poisson kernel for the BSDF, and a reflectance derived from the Robin coefficient for the albedo.

Participating Media. Some of the most striking images rendering produces—clouds, smoke, fog, marble, skin—come from media with no surfaces at all, made of far more microscopic particles than any computer could store. Rendering simulates them without ever materializing one. Inside such a medium, the geometry a ray encounters is the stochastic particle geometry of Section 2.5, and radiance becomes a random variable: where a surface returns a unique hit, the medium returns a free-flight sample. Volume rendering asks only for the expectation [Novák et al. 2018],

$$L(x, -\vec{\omega}) = \mathbb{E}_{t \sim p_{x, \vec{\omega}}^{\text{ff}}} [L(r_{x, \vec{\omega}}(t), -\vec{\omega})]. \quad (2.25)$$

Expanding the expectation over its two outcomes—collide with a particle before the nearest surface at distance z , or fly past it—recovers the *volume rendering equation*,

$$L(x, -\vec{\omega}) = \int_0^z p_{x, \vec{\omega}}^{\text{ff}}(t) L_s(r_{x, \vec{\omega}}(t), -\vec{\omega}) dt + T(z) L(r_{x, \vec{\omega}}(z), -\vec{\omega}), \quad T(z) := \int_z^\infty p_{x, \vec{\omega}}^{\text{ff}}(t) dt, \quad (2.26)$$

where L_s collects light in-scattered at the collision, and the transmittance T is nothing but the tail probability of the free-flight distribution (2.19)—the same law sampled in Section 2.5, with delta tracking as the heterogeneous sampler. Chapter 4 replays exactly this move for WoS: the solution over a random medium becomes an expectation over sampled closest particles, expanded one query at a time.

Virtual Point Lights. Interactive applications want global illumination—light bouncing through a scene—at real-time rates, a budget no fresh-paths-per-pixel estimator can meet. The answer is reuse. A path tracer discards every path after one use; Keller [1997] instead traces a set of paths from the lights, deposits their vertices as N *virtual point lights* with positions y_i and powers Φ_i , and lets every shading point gather from all of them,

$$L(x, \vec{\omega}) \approx \sum_{i=1}^N f(x, \vec{\omega}, x\vec{y}_i) \mathcal{G}(x, y_i) V(x, y_i) \Phi_i, \quad (2.27)$$

where the geometry term $\mathcal{G}$ and visibility V connect each light to each shading point. One set of samples now serves every evaluation point—an idea that grew into a whole family of many-light methods [Dachsbacher et al. 2014]—at the cost of correlated error and a singular kernel $\mathcal{G}$ that must be clamped, with the clamped energy restored by a separate correction step [Kollig and Keller 2006]. The caching scheme of Chapter 5 is the elliptic analogue: boundary samples shared across all interior points, with the Green’s function and Poisson kernel in the role of $\mathcal{G}$ —singularities, clamping, and all.

Differentiable Rendering. Each capability so far is a forward simulation; differentiable rendering inverts them, recovering the materials, lighting, or shape that produced an image. Optimizing through a renderer requires derivatives of images with respect to scene parameters, and a Monte Carlo estimator supplies these as well. For parameters π that leave the geometry fixed—medium and material coefficients—one can differentiate the rendering equation (2.24) directly, and the result is an equation of the same recursive form: differential radiance scatters

through the unchanged BSDF, driven by a new emission term assembled from the primal radiance [Gkioulekas et al. 2013, 2016; Zhang et al. 2019; Nimier-David et al. 2020],

$$\frac{\partial L}{\partial \pi}(x, \vec{\omega}) = \int_{S_+^2} f(\langle \vec{n}, \vec{\omega}' \rangle) \frac{\partial L}{\partial \pi}(r_{x, \vec{\omega}'}^*, -\vec{\omega}') d\vec{\omega}' + \underbrace{\frac{\partial L_e}{\partial \pi} + \int_{S_+^2} \frac{\partial f}{\partial \pi}(\langle \vec{n}, \vec{\omega}' \rangle) L(r_{x, \vec{\omega}'}^*, -\vec{\omega}') d\vec{\omega}'}_{\text{a new emission term}}, \quad (2.28)$$

so the very same path tracer estimates derivatives, fed a different emission. Geometric parameters are harder: they *move* the visibility discontinuities, and differentiating under the integral misses them entirely—on either side of a silhouette the integrand is locally constant in π , so the interior derivative records nothing as the silhouette sweeps past. Reynolds transport supplies the missing contribution—the jump across the discontinuity times its normal velocity—first analyzed for shadows by Ramamoorthi et al. [2007] and turned into an estimator by Li et al. [2018]. Writing a pixel value as an integral $\int_X m(x; \pi) dx$ of its measurement integrand m —the filter-weighted radiance carried through screen position x , not to be confused with the BSDF—they sample the moving discontinuity set $E(\pi)$ directly:

$$\frac{d}{d\pi} \int_X m(x; \pi) dx = \int_X \frac{\partial m(x; \pi)}{\partial \pi} dx + \int_{E(\pi)} (m^-(x; \pi) - m^+(x; \pi)) V_{\vec{n}}(x) dx. \quad (2.29)$$

Chapter 6 inherits both halves in differential form. The derivative of a PDE solution satisfies a boundary value problem of the same class, whose boundary data plays the role of the new emission; and because the moving “discontinuity” is the domain boundary itself, its motion enters that data pointwise—as the normal velocity $V_{\vec{n}}$ times the solution’s normal derivative—Reynolds transport expressed as a boundary condition rather than an edge integral.

In every stop on this tour, the pattern is the one assembled in Sections 2.1–2.5: write the quantity of interest as a recursive integral equation, estimate it with single-sample recursion over sampled geometry, and pay for generality with variance rather than discretization. The chapters that follow replay this playbook, replacing radiance with the solutions of elliptic boundary value problems.

Chapter 3

First-Order Linear Boundary Conditions

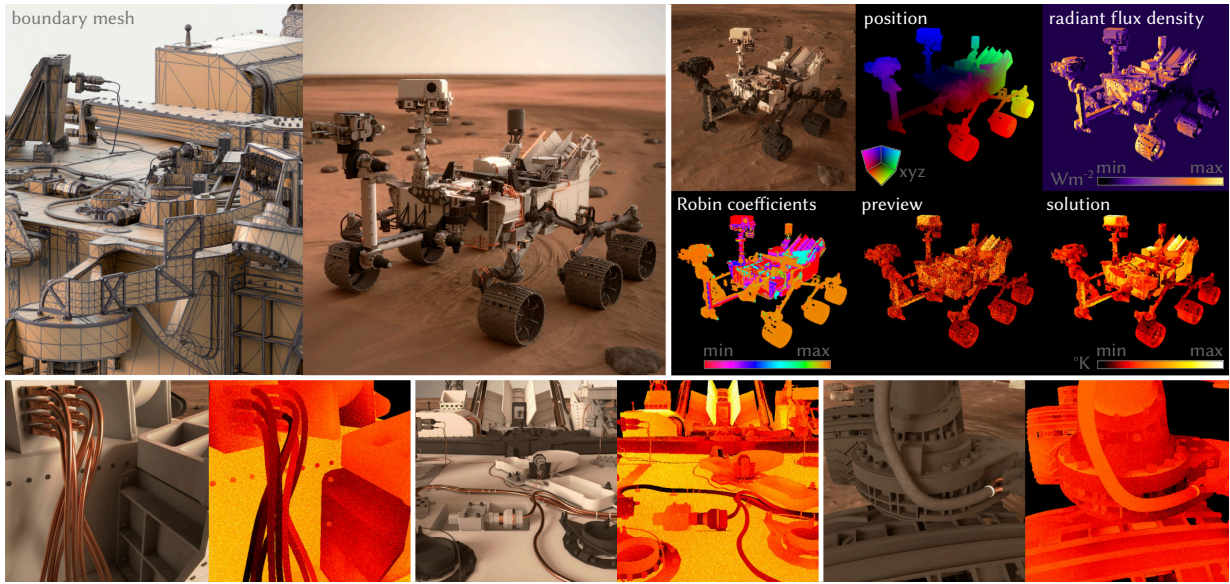


Figure 3.1: Thermal analysis of NASA’s Curiosity rover with walk on stars (WoSt): a Laplace equation with Robin boundary conditions solved directly on an artist-created model that state-of-the-art tetrahedral meshers cannot robustly process. Deferred shading launches walks only at visible points, with sample density proportional to screen coverage (Section 3.4).

This chapter develops a recursive Monte Carlo estimator for linear elliptic PDEs with general first-order linear boundary conditions. We begin by considering the Dirichlet boundary condition, which prescribes the exact value of the solution on the boundary,

$$\begin{aligned}\Delta u(x) &= 0, & x &\in \Omega, \\ u(x) &= g(x), & x &\in \partial\Omega.\end{aligned}\tag{3.1}$$

These are perhaps the simplest boundary conditions we can impose. In rendering, the Dirichlet condition would be analogous to prescribing the outgoing radiance at every surface. WoSt handles this case naturally (Section 2.3)—the value upon terminating a random walk in the

ε -shell is evaluated directly as $u(x) = g(x)$. Yet a fixed boundary value rarely models the physical situation we care about.

More often we want to model a *behavior* or response at the boundary rather than a fixed value. Robin conditions enable this modeling by constraining the relationship between the solution and its normal derivative,

$$\begin{aligned} \Delta u(x) &= 0, & x &\in \Omega, \\ \frac{\partial u(x)}{\partial \vec{n}_x} - \mu(x) u(x) &= h(x), & x &\in \partial\Omega. \end{aligned} \quad (3.2)$$

In thermal modeling, for instance, the relationship between the temperature and its gradient at the boundary determines the rate of heat exchange with a surrounding ambient medium. Therefore, Robin boundary conditions can be thought of as the analogue of a material model, since they model behavior rather than prescribe fixed values (Figure 1.2). Throughout this chapter we require the Robin coefficient to be non-negative, $\mu \geq 0$; negative coefficients model emissive boundaries that inject rather than absorb energy, and fall outside the scope of the estimators we develop. Unfortunately, WoS struggles to incorporate Robin boundary conditions in a way that preserves geometric scalability.

To understand why Robin conditions are so challenging for WoS, it helps to first revisit how this class of estimators is motivated. Sawhney and Crane [2020] present two complementary derivations—a stochastic one, identifying the solution with the exit value of a Brownian walk, and an integral one, the recursive mean value property. Under the stochastic perspective WoS is an accelerated simulation of Brownian motion, where rather than advancing by many small steps, as in Euler–Maruyama, the walk takes a single large step across a maximal empty sphere at each iteration. Later work preserves these large steps even when the underlying process carries spatially varying drift, diffusion, or absorption [Sawhney et al. 2022].

The acceleration is therefore an interior phenomenon, and it breaks down at the boundary. For Dirichlet conditions this costs nothing: the maximal balls simply shrink as the walk nears $\partial\Omega$, and the walk terminates once it enters the ε -shell. For Robin conditions, existing extensions instead reflect off the boundary [Mascagni and Simonov 2004; Maire and Tanré 2012; Zhou et al. 2015], taking a small offset back into the domain—and the estimator collapses into exactly the small-step random walk the acceleration was meant to avoid (Figure 3.4, left). The integral viewpoint suggests a way out: rather than reflect, one can write a mean value relation directly at the boundary, as prior work has done for restricted geometries and specific conditions [Ermakov and Sipin 2009; Simonov 2017].

We take this integral route in full generality, preserving the efficiency of WoS under general boundary conditions and bringing PDE solving closer to rendering. Where prior work develops the estimator through the stochastic lens and handles the boundary by simulating reflections, we make the integral representation the primary object and derive a single recursive equation that treats first-order conditions directly. That representation is

Walk on stars

$$\alpha(x)u(x) = \int_{\partial St(x,s)} \rho_\mu(x,y) P(x,y) u(y) dy - \int_{\partial St_R(x,s)} G(x,y) h(y) dy \quad (3.3)$$

which expresses u at a point as a recursive integral over the boundary of a star-shaped region $St(x, s)$, plus a source term carrying the forcing of the Robin condition. The factor $\alpha(x)$ is position-dependent and equals one in the interior; we treat it fully in Section 3.1. Taken together, three properties of this representation mirror the structure of Monte Carlo rendering:

- it is a direct analogue of the rendering equation, with ρ_μ playing the role of an albedo, the Poisson kernel P a geometry term, a star-shaped subdomain ∂St serving as the visible portion of a scene, and the Green's function weighted Robin condition Gh an emission term;
- it can be realized by a single-sample recursive Monte Carlo estimator that shares the same kernel sampling method—a directional sampling or ray-casting step—and Russian roulette termination used by rendering algorithms; and
- its convergence can be established by the same operator-theoretic arguments that justify recursive estimators in rendering.

The core of the chapter develops this representation in full. We derive (3.3) from the boundary value problem (3.2) (Section 3.1), construct a recursive Monte Carlo estimator together with the conditions that make it converge (Section 3.2), and place that convergence on rigorous footing through operator theory (Section 3.3).

3.1 Recursive Boundary Integral Equation

The first step toward a recursive Monte Carlo estimator is to express u as an integral that satisfies the PDE of (3.2) by construction and that depends on u only through its own values. This recursive dependence makes the representation a Volterra integral equation, and lets us solve the PDE by repeatedly estimating that integral with a Monte Carlo estimator, exactly as rendering repeatedly estimates the rendering equation.

Encoding the Robin Condition. Our starting point is Green's third identity (2.8) on a position-dependent subdomain $A(x) \subset \Omega$: any u of this form satisfies $\Delta u = 0$ by construction, for any boundary conditions imposed on G . Unlike WoS, the recursion in a Robin problem evaluates u on the boundary itself, where $\alpha = \frac{1}{2}$. Two obstacles separate the identity from a recursive estimator: it does not yet encode the Robin condition, and it depends on the unknown normal derivative $\partial u / \partial \vec{n}$, which—left in place—would spawn a cascade of branching derivatives of increasingly high order. A single regrouping addresses both at once. Adding and subtracting μGu on $\partial A(x)$ gives

$$\alpha(x) u(x) = \int_{\partial A(x)} [P(x, y) - \mu G(x, y)] u(y) - G(x, y) \left[\frac{\partial u(y)}{\partial \vec{n}_y} - \mu u(y) \right] dy, \quad (3.4)$$

the real-valued analogue of the Brakhage–Werner trick [Nédélec 2001]. The second bracket is now exactly the left-hand side of the Robin condition in (3.2), and the unknown normal derivative appears nowhere else.

This regrouping removes the normal derivative in two complementary ways. Where $\partial A(x)$ coincides with the true boundary, $x \in \partial A(x) \cap \partial \Omega$, the second bracket is known data: the Robin

condition fixes $\partial u / \partial \vec{n}_y - \mu u = h$. Everywhere else, we are free to choose G , and we choose it to vanish on the artificial part of the boundary, $G = 0$ on $\partial A(x) \setminus \partial \Omega$, annihilating the term wherever the data is unknown. A convenient choice is the Green's function of a ball, which is known in closed form (Equation 2.9). Taking the subdomain $St(x, s)$ to be the connected component of $B(x, s) \cap \Omega$ containing x leads the Robin term to vanish everywhere except on the true boundary, where it is known. The identity collapses to

$$\alpha(x) u(x) = \int_{\partial St(x, s)} [P(x, y) - \mu G(x, y)] u(y) dy - \int_{\partial St_R(x, s)} G(x, y) h(y) dy, \quad \partial St_R := \partial St \cap \partial \Omega. \quad (3.5)$$

A Recursive Integral Equation. Factoring P out of the recursive term casts (3.5) in a more suggestive form,

$$u(x) = \int_{\partial St(x, s)} \rho_\mu(x, y) \frac{P(x, y)}{\alpha(x)} u(y) dy + \frac{H(x)}{\alpha(x)}, \quad \rho_\mu(x, y) := 1 - \mu \frac{G(x, y)}{P(x, y)}, \quad (3.6)$$

where ρ_μ is a reflectance factor encoding the Robin coefficient and $H(x) := - \int_{\partial St_R} G(x, y) h(y) dy$ collects the source contribution over ∂St_R . This is the representation previewed in (3.3): it satisfies both the Laplacian and the Robin condition by construction, depends on u only recursively through its boundary values, and—being Volterra—is amenable to recursive Monte Carlo estimation. One choice remains open, however: the radius s of the subdomain, which we have so far left free. The next section constructs the estimator and shows that convergence constrains it to a specific admissible value.

3.2 Walk on Stars

The Volterra integral equation from the previous section can be estimated by a single-sample Monte Carlo estimator that estimates $u(x_0)$ by repeatedly accumulating an additive contribution from H and then drawing x_{k+1} from a density $p(\cdot | x_k)$ on ∂A ,

$$\hat{u}(x_k) = \frac{\rho_\mu(x_k, x_{k+1}) P(x_k, x_{k+1})}{\alpha(x_k) p(x_{k+1} | x_k)} \hat{u}(x_{k+1}) + \hat{H}(x_k). \quad (3.7)$$

The resulting sequence of points $x_0, x_1, x_2, \dots$ forms a path, analogous to the light paths traced in rendering algorithms. This is nearly a complete estimator, but on subdomains formed from spheres of arbitrary radius the recursion neither converges nor offers a natural termination criterion. Unrolling it makes the requirement precise: a path starting at x_0 accumulates an infinite series of contributions,

$$\hat{u}(x_0) = \sum_{k=0}^{\infty} \left(\prod_{i=0}^{k-1} \frac{\rho_\mu(x_i, x_{i+1}) P(x_i, x_{i+1})}{\alpha(x_i) p(x_{i+1} | x_i)} \right) \hat{H}(x_k), \quad (3.8)$$

which converges with finite variance only when the per-step throughput $\rho_\mu P / (\alpha p)$ stays bounded below one. As in rendering, a contracting throughput lets us either truncate paths at finite depth with controlled bias or terminate them probabilistically with Russian roulette. We bound the two factors of the throughput in turn.

The Walk on Stars Estimator. Setting $p(y | x) = P(x, y)/\alpha(x)$ collapses $P/(\alpha p)$ to one, since P integrates to $\alpha(x)$ over ∂A and P/α is therefore its normalized density. The Poisson kernel is the signed solid-angle density viewed from x [Jacobson et al. 2013], positive on visible points and negative on occluded ones, so requiring it to be a valid density is a geometric condition: A must be *star-shaped* with respect to x , every boundary point visible along a single ray (Figure 3.2, left). We write $St(x, s)$ for such a domain of radius s centered at x .

The remaining factor is the reflectance product $\prod_i \rho_\mu(x_i, x_{i+1})$, which we bound by

$$|\rho_\mu(x, y)| = \left| 1 - \mu \frac{G(x, y)}{P(x, y)} \right| \leq 1, \quad (3.9)$$

enforced by shrinking the radius of the subdomain $St(x, s)$ so that ∂St_R does not contain any point whose reflectance is not bounded to the unit interval $\rho_\mu \in [0, 1]$. Without this restriction the reflectance ranges over $(-\infty, 1)$: not only can its magnitude grow without bound, its sign can flip from step to step, so the product along a path suffers catastrophic cancellation. Restricting ρ_μ to $[0, 1]$ rules out both failure modes, and—as we will see when terminating walks—lets the reflectance double as a survival probability.

We refer to the resulting estimator as *walk on stars* (WoSt), since the walk occurs over star-shaped subdomains that have been selected to bound the path throughput. Writing $\hat{H}(x_k)$ for the single-sample source contribution over ∂St and $\xi \sim \mathcal{U}(0, 1)$, the WoSt estimator can be expressed in a simple form

$$\hat{u}(x_k) = \hat{H}(x_k) + \begin{cases} \hat{u}(x_{k+1}), & \xi < \rho_\mu(x_k, x_{k+1}), \\ 0, & \text{otherwise.} \end{cases} \quad (3.10)$$

Implementing this algorithm only requires efficiently computing star-shaped domains at each step of the walk, sampling the Poisson kernel over the star-shaped domain, and deciding when to terminate. The remainder of this section describes these three operations in turn: how to choose the star-shaped domain, how to sample the Poisson kernel on its boundary, and how to terminate the resulting walk.

Constructing Star-Shaped Domains. The estimator requires computing, at each step, the largest radius s for which $St(x, s)$ stays star-shaped and keeps $\rho_\mu \in [0, 1]$ (Figure 3.2). The first condition caps s at the distance to the closest silhouette point of the boundary seen from x ; the second caps it further wherever the Robin coefficient is large.

To make the second cap concrete, we substitute the closed-form 3D Green’s function and Poisson kernel of a ball $B(x, s)$ (Equation 2.9) into the reflectance ρ_μ of Equation (3.6). For any $y \in \partial St_R$, this gives

$$\rho_\mu(x, y) = 1 - \frac{\mu(y) r}{\cos \theta} \left(1 - \frac{r}{s} \right), \quad (3.11)$$

where $r = \|y - x\|$ and $\cos \theta = \vec{n}_y \cdot (y - x)/r$. Both $1 - r/s$ and $\cos \theta$ are positive—the former since $r \leq s$, the latter since ∂St is front-facing by construction—so $\rho_\mu \in [0, 1]$ rearranges to an

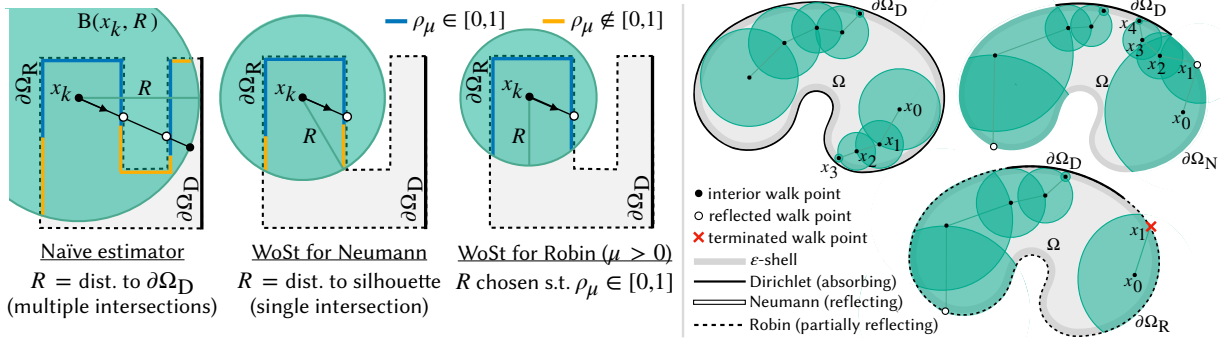


Figure 3.2: (Left) Selecting the star radius. A naïve radius capped only by the absorbing boundary admits occluded points, where the Poisson kernel is no longer a valid density; restricting to the closest silhouette restores a star-shaped region ($\mu = 0$), and for $\mu > 0$ the radius shrinks further to keep $\rho_\mu \in [0, 1]$ —as $\mu \rightarrow \infty$ it collapses to the boundary distance, recovering absorbing behavior. (Right) The resulting estimator handles arbitrary mixtures of boundary conditions with a single walk: walks land on reflecting and partially reflecting boundaries and continue, surviving Russian roulette with probability ρ_μ , and terminate in the ε -shell of absorbing boundaries.

upper bound on the radius,

$$s \leq \frac{r}{1 - \frac{\cos \theta}{\mu(y) r}} \quad \text{when } r > \frac{\cos \theta}{\mu(y)}, \quad (3.12)$$

which must hold at every $y \in \partial St_R$. On the spherical part $\partial St \setminus \partial St_R$ the Green's function vanishes, so $\rho_\mu = 1$ and no constraint on s arises.

As μ ranges from 0 to ∞ , the admissible radius interpolates continuously between the distance to the nearest silhouette and the distance to the boundary itself—a more absorbing boundary admits smaller steps, matching the intuition that a Brownian motion travels less far before being absorbed. Both caps are local predicates on faces and edges, so a single hierarchy traversal—accelerated by a bounding volume hierarchy augmented with normal cones [Johnson and Cohen 2001]—returns the largest admissible s , which we clamp below by a minimum radius $s_{\min}$ to keep walks from stalling near concave boundaries. We tie this clamp to the same tolerance ε that defines the Dirichlet ε -shell (Section 2.3), setting $s_{\min} = \varepsilon$, so that a single parameter ε governs both the shell in which absorbing walks terminate and the smallest star a reflecting walk may take. This clamp is not free: near concave portions of the reflecting boundary it can render St no longer star-shaped, and the estimator then effectively assumes $u = 0$ on the portion of ∂St hidden from x . Like the ε -shell that terminates WoS on Dirichlet boundaries, whose bias and cost are well understood [Binder and Braverman 2012; Sawhney and Crane 2020], $s_{\min}$ is therefore a source of controlled bias, trading accuracy for shorter walks. Figure 3.3 studies this tradeoff empirically: overly large values of ε appear as a global darkening of the solution where boundaries predominantly reflect, and as a Voronoi-like flattening where they absorb; in practice the tradeoff is forgiving, and a single default serves all our experiments.

This radius query is the one geometric operation WoSt adds beyond rendering's standard ray-intersection queries. It behaves like a visibility query—returning, in effect, how far the walk

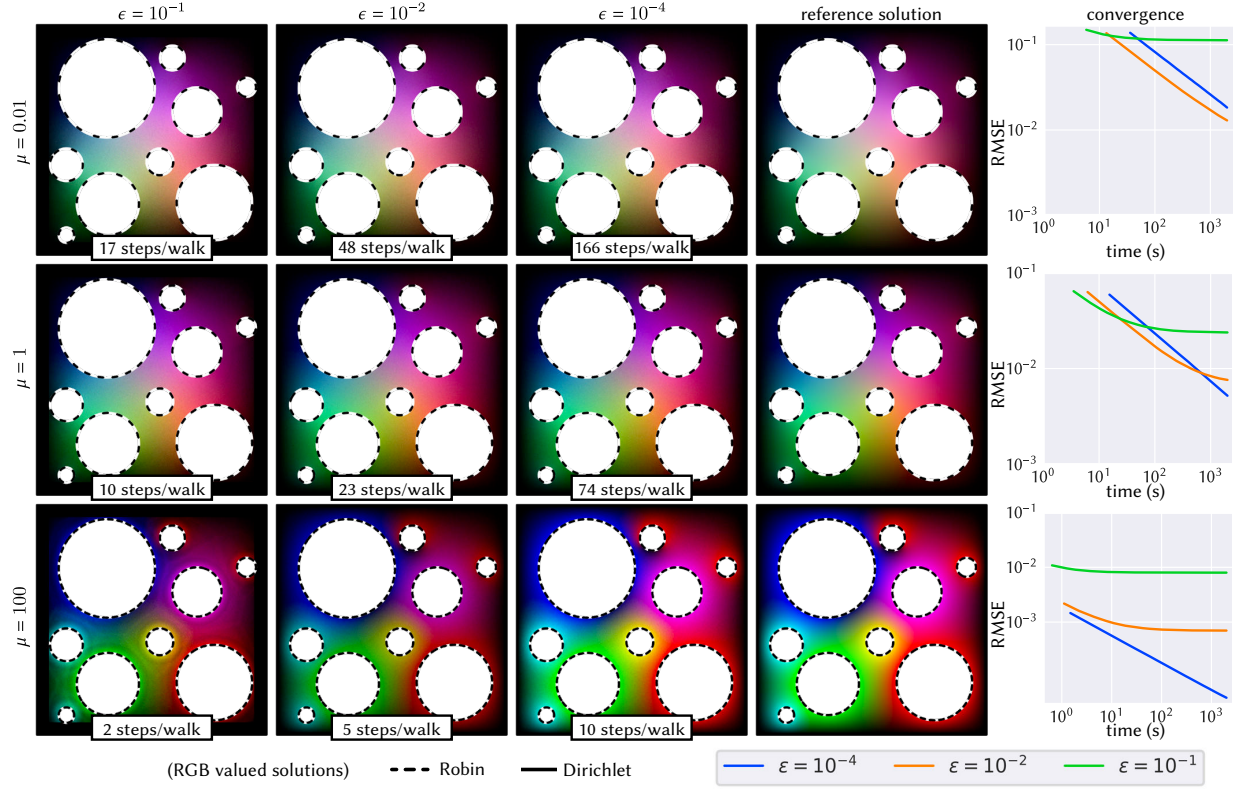


Figure 3.3: A single ϵ tolerance controls the tradeoff between bias and efficiency. On predominantly reflecting Robin boundaries, overly large ϵ manifests as a global darkening of the estimated solution (top two rows); on absorbing boundaries it instead produces Voronoi-like flattening (bottom row).

can see before the boundary either occludes it or, through the reflectance bound, absorbs too much of its throughput.

Sampling Star-Shaped Domains. Given the subdomain, we sample the next point $x_{k+1} \in \partial St$ proportional to P/α . The key observation, shared with Monte Carlo rendering, is that the Poisson kernel of a ball is exactly the solid-angle density subtended by ∂St at x [Jacobson et al. 2013]. Sampling it therefore requires no explicit evaluation of P at all: we cast a ray from x in a uniformly sampled direction and take x_{k+1} to be its first intersection with ∂St , exactly as a path tracer samples an outgoing direction and intersects the scene. This importance samples P/α perfectly, canceling it from the throughput and leaving ρ_μ as the sole per-step factor. When x lies on a Robin boundary we sample only the inward hemisphere about $-\vec{n}_x$; the resulting $\frac{1}{2}$ cancels the $\alpha = \frac{1}{2}$ in the denominator and guarantees $x_{k+1} \in \Omega$. As in ray tracing, we offset the ray origin slightly inward before casting for numerical robustness, avoiding spurious self-intersections with the boundary the walk currently sits on.

Terminating Walks. The reflectance $\rho_\mu \in [0, 1]$ doubles as a survival probability, so we terminate walks by Russian roulette: at each step the walk continues with probability $\rho_\mu(x_k, x_{k+1})$ and otherwise stops, with no reweighting on survivors. This holds the throughput at one—just

as a path tracer prunes low-throughput paths—and lets walks terminate directly on the Robin boundary, with more absorbing boundaries (larger μ) ending walks sooner.

In cases where $\mu = 0$, the reflectance is identically one and no walk is ever terminated by Russian roulette. Recovering a unique solution in these cases requires Tikhonov regularization, which replaces the Laplacian with a screened operator $\Delta - \sigma$ and its corresponding Green's function and Poisson kernel (Equation 2.13). In rendering, this would amount to simulating light transport on a room that consisted entirely of mirrors (Figure 1.2, right).

3.3 Operator-Theoretic Convergence

As Monte Carlo rendering was gaining interest in the late 1990s, Arvo [1995] and Veach [1998] established formal convergence guarantees for rendering. It is now largely taken for granted that one can throw geometry, materials, and lighting at a renderer and trust that enough samples will eventually resolve the image. Yet this guarantee is the basis on which many important developments in rendering over the past three decades rely. Without finite-variance guarantees, the $O(1/\sqrt{n})$ convergence rate of Monte Carlo estimation (2.3) would not hold, and efforts to reduce variance or accelerate the estimator through caching would be for naught. We establish the analogous guarantee for the WoSt algorithm, making the intuitive arguments of the previous section rigorous through operator theory.

A Rendering-Style Convergence Argument. We adapt Arvo [1995]'s operator analysis of the rendering equation to this setting. Arvo [1995, Theorems 13–14] factors the light-transport operator into a *local reflection operator* $\mathcal{K}$, whose kernel is the bidirectional reflectance distribution function (BRDF), composed with a *field-radiance operator* $\mathcal{G}$ that carries the geometry and visibility of transport between surfaces, and bounds the spectral radius by bounding the two factors separately. This kernel admits the same decomposition: the reflectance term ρ_μ plays the role of $\mathcal{K}$ and the Poisson kernel P that of $\mathcal{G}$. On $L^2(\Omega)$, Equation (3.6) takes the second-kind operator form of Section 2.2,

$$u = C_{\rho_\mu P/\alpha}^{St} u + H/\alpha, \quad (C_\kappa^A u)(x) := \int_{\partial A(x)} \kappa(x, y) u(y) dy, \quad (3.13)$$

and classical results [Heinrich and Mathé 1993; Jörgens 1982] guarantee a bounded resolvent and finite-variance recursive Monte Carlo estimator whenever the spectral radius $r(C_{\rho_\mu P/\alpha}^{St}) < 1$. A Hölder-type inequality factors the spectral radius bound as

$$\sup_x \int_{\partial St} |\rho_\mu P/\alpha| dy \leq (\sup |\rho_\mu|) \left(\sup_x \int_{\partial St} |P/\alpha| dy \right), \quad (3.14)$$

and these two geometric conditions handle each factor: star-shapedness gives $\int |P/\alpha| dy = 1$, and the trimming construction gives $|\rho_\mu| \leq 1$, so the product is at most one. This mirrors Arvo [1995]'s argument: his Theorem 13 bounds the reflection operator by $\|\mathcal{K}\| < 1$ whenever the directional-hemispherical reflectance is bounded away from one, while the field-radiance operator satisfies $\|\mathcal{G}\| \leq 1$ because radiance is conserved along free-space transport.

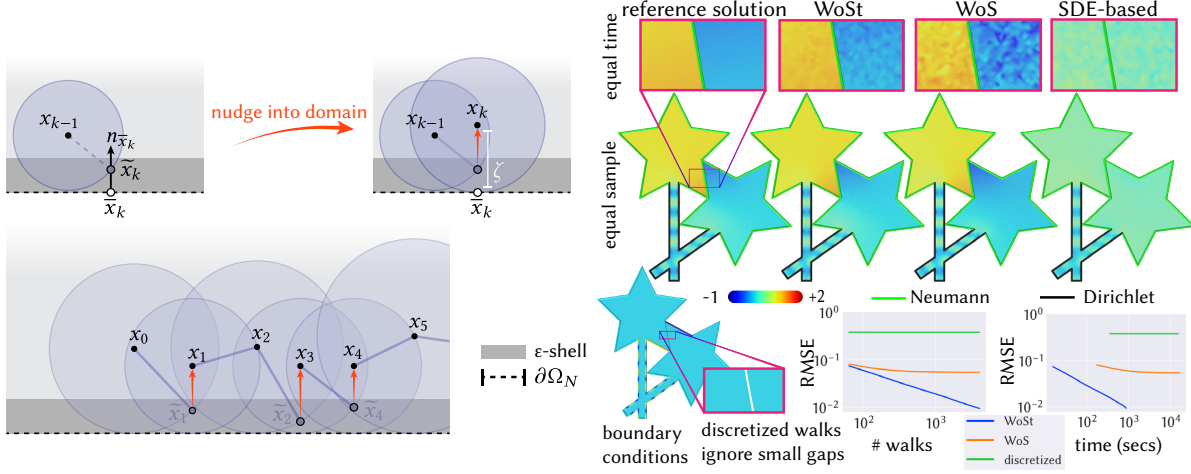


Figure 3.4: (Left) Prior methods simulate reflections by nudging walks back into the domain by a fixed offset ζ , so walks take many small steps near reflecting boundaries. (Right) At comparable parameter settings, WoSt is both faster and less biased than WoS with discretized reflections or estimators based on stochastic differential equations (SDEs): where discretized walks plateau at a bias floor—ignoring geometry below their offset scale—WoSt continues to converge at the expected Monte Carlo rate.

Resolvent Interpretation. With both factors bounded, the spectral radius satisfies $r < 1$ and the resolvent $(I - C_{\rho_\mu P/\alpha}^{St})^{-1}$ exists. Unrolling it as a Neumann series (2.5) recovers the path sum of Equation (3.8), so the recursive walk is just a stochastic evaluation of this inverse. The strict bound $r < 1$ fails only in the limiting cases $\mu \rightarrow \infty$ (pure Dirichlet, a perfectly absorbing boundary that reflects nothing back) and $\mu \equiv 0$ (pure Neumann, a perfectly reflecting boundary with $|\rho_\mu| = 1$). But as Veach [1998, Section 4.4] notes, per-step contraction is stronger than needed: as in Section 2.2, the series converges whenever $\|C^i\| < 1$ for some $i \geq 1$, so individual steps may graze a perfectly reflecting boundary without destroying convergence, provided energy is dissipated over finitely many steps. When it is not, Tikhonov screening (Neumann) restores strict contraction.

3.4 Evaluation

We evaluate WoSt on three fronts: against the finite-difference stencils that previously handled Neumann conditions, against the walk-on-boundary family of grid-free estimators, and on geometry beyond the reach of mesh-based solvers.

Discretized Reflections. Prior grid-free treatments of Neumann conditions handle reflection by offsetting walks near the boundary by a fixed distance ζ along the inward normal, and accounting for the boundary data through a finite-difference correction tied to that offset [Mascagni and Simonov 2004; Maire and Tanré 2012; Zhou et al. 2015] (Figure 3.4, left). The offset ζ is bad in both directions: large values inject bias—geometry smaller than the offset is simply ignored—while small values cause walks to stall against the boundary, hurting efficiency.

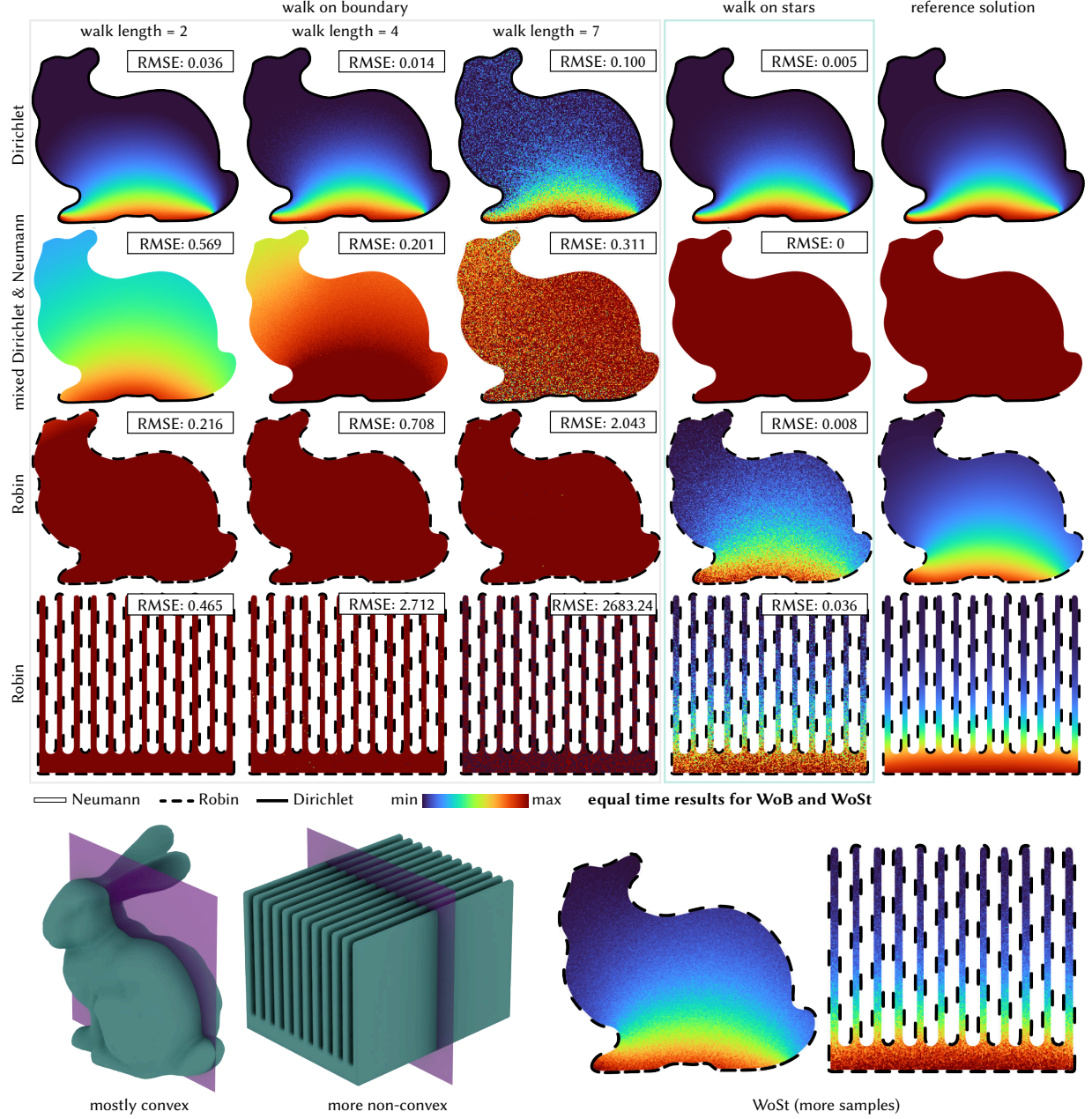


Figure 3.5: Walk on boundary suffers an extreme bias–variance tradeoff in nonconvex domains: truncating walks introduces severe bias, while untruncated walks exhibit exponential variance growth. WoSt instead converges reliably at the expected Monte Carlo rate for any combination of Dirichlet, Neumann, and Robin conditions.

Estimators based on reflecting SDEs fare even worse, accumulating discretization bias at every step rather than only at the boundary. WoSt instead steps in a single large jump directly off the boundary, eliminating the offset entirely. The resulting gap is not a constant factor that more samples can close: at equal time WoSt is roughly an order of magnitude more accurate,

and where discretized walks plateau at a bias floor set by their offset, WoSt converges at the expected Monte Carlo rate (Figure 3.4, right).

Walk on Boundary. Walk on boundary (WoB) [Sabelfeld and Simonov 1994; Sugimoto et al. 2023] was likewise developed to avoid finite-difference approximations, and is the grid-free method closest to path tracing, relying on ray intersection alone. As discussed in Section 2.4, its estimator has unbounded variance on nonconvex geometry. Figure 3.5 shows this numerically. As the domain becomes more nonconvex, WoB’s error stops falling at the expected rate even with an optimized GPU implementation, while WoSt converges at the usual $O(1/\sqrt{N})$ rate across all domains. Variance reduction and faster hardware shrink the constant on that rate, but cannot rescue an unbounded estimator.

Large-Scale Geometry. This combination of bounded convergence and scalability means that, just like rendering, WoSt can be applied directly to large, complex scenes without concern for mesh quality or complexity. We demonstrate this robustness with a thermal analysis of NASA’s Curiosity Mars rover (Figure 3.1), solving a Laplace equation with Robin conditions on an artist model on which state-of-the-art tetrahedral meshing destroys detail or exhausts memory, costing up to a day before any solve (Figure 1.3, left). WoSt instead evaluates the solution pointwise, with no mesh or global system. Crucially, this also makes evaluation output-sensitive: to visualize the rover’s thermal behavior we run the solver only on the visible surfaces, with sample density proportional to screen-space area, so parts of the rover covering more pixels receive more samples. Deferred shading launches walks only at visible points, giving progressively refined estimates in minutes rather than days. The toaster and lung simulations of the introduction (Figures 1.1 and 1.4) are further examples of this scalability, each evaluated directly on high-resolution geometry on which volumetric meshing fails or is impractical.

3.5 Outlook

The recipe of this chapter is general: write the PDE as a recursive integral equation, condition the operator to have spectral norm below one, and come up with a sampling procedure for the kernel. Several directions remain open, spanning the geometry the estimator runs on, the operators it can represent, and the boundary conditions it can encode.

Geometric Representations. The most immediate limitation is geometric. Unlike the closest-point query of the original WoS [Muller 1956; Sawhney and Crane 2020], or the ray-intersection query of a renderer [Pharr et al. 2023], the star-shaped query of Section 3.2 is comparatively involved: it must locate the closest visibility silhouette and simultaneously enforce the reflectance bound. We have shown how to evaluate it on triangle meshes, but extending it to the richer geometric representations that grid-free methods otherwise handle gracefully—implicit surfaces, spline patches, instanced or procedural geometry—is nontrivial precisely because the query couples visibility with a per-point coefficient bound. Recent work [Huang 2025] suggests this is feasible, although efficiency of these queries on alternative representations is still a bottleneck.

Broadening the set of representations on which the star-shaped query can be computed *efficiently* is, at present, the main obstacle to applying WoSt as widely as WoS.

Beyond Scalar Elliptic Operators. The conditioning that makes the integral representation (3.3) convergent assumes a linear elliptic setting; whether the same trick—casting the solution as a Volterra integral and bounding its operator—extends beyond it is genuinely unclear. Operators such as Helmholtz, with notoriously large spectral radii, suggest a real limit. Even within the linear elliptic regime, vector-valued problems such as linear elasticity and Stokes flow remain out of reach: their components mix across dimensions, and while one might hope to simply decompose them into separate scalar harmonic PDEs, the constituent solutions recouple at the boundary without a clear means of termination. Yu et al. [2025] take a first step in broadening the regime, conditioning the integral equation through integration by parts to estimate spatial derivatives $\partial u / \partial x$ of a Poisson equation directly. Where such conditioning transformations cannot be found, one might instead use WoSt as an inner solve within a nested, coupled, or iterative scheme, much as mesh-based methods build solvers for involved PDEs atop simpler linear ones [Rioux-Lavoie et al. 2022; Jain et al. 2024; Bati et al. 2023].

Nonlinear Boundary Behavior. Boundary conditions can be considered beyond the first-order linear regime. The radiative transfer that motivates the thermal analysis example in Section 3.4 is in fact nonlinear in temperature T —the Stefan–Boltzmann flux scales as T^4 —and we, like Bati et al. [2023], linearize it into an effective Robin coefficient. This nonlinearity breaks the linear operator structure on which WoSt’s convergence analysis rests. The natural recourse is precisely the subroutine pattern described above: Bao et al. [2026] wrap a Robin WoSt solver in a Picard-style fixed-point iteration, resolving the nonlinear radiative coupling that a single linearization cannot. Their scheme is only empirically convergent, however—strict contraction is not guaranteed—leaving open whether nonlinear boundary behavior admits a directly contractible integral representation.

Chapter 4

Participating Media

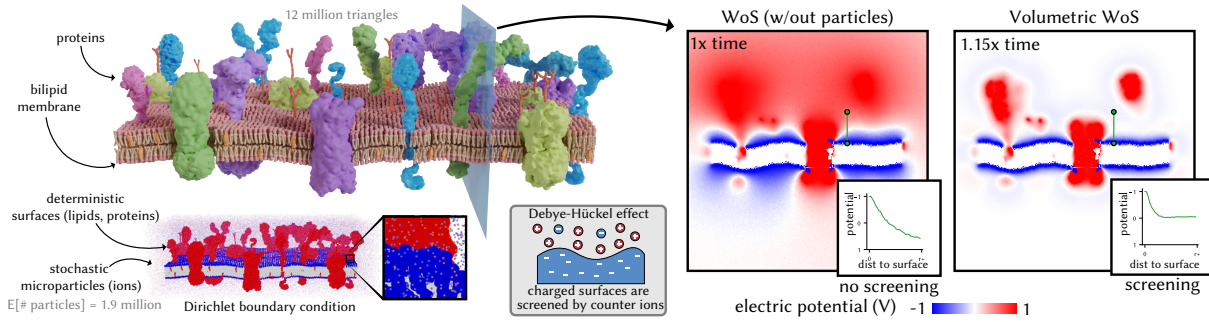


Figure 4.1: Physical systems such as biological membranes involve extraordinarily complex microparticle geometry—here, ions in solution mediating electrostatic interactions. Volumetric walk on spheres estimates the mean solution of a PDE over such geometry while realizing only the particles each walk encounters (Section 4.5).

Our goal in this chapter is to develop Monte Carlo methods that efficiently and accurately simulate physical effects on complex *microparticle geometry*. We focus on *participating media*: microscopic particles distributed throughout a larger domain and described statistically rather than by an explicit configuration. Examples include water droplets in clouds, ions in solution around a cellular membrane, and inclusions in porous or colloidal materials. Though small relative to the domains they occupy, these particles strongly influence the quantities of interest: cloud droplets affect radiative transfer [Chandrasekhar 2013] and photochemical reactions [Faust 1994; Bianco et al. 2020], while ions mediate electrostatic screening near biological membranes [Andelman 1995]. Such systems remain difficult to analyze with conventional numerical methods because simulation appears to require resolving a vast configuration of microscopic geometry.

The difficulty is not primarily one of modeling the physics. The relevant phenomena are often described by standard PDEs with appropriate boundary conditions; the challenge is that the boundary geometry depends on a random particle configuration $\mathcal{O}$. If $u_{\mathcal{O}}$ denotes the solution for one such configuration, the natural objective is the mean solution

$$\bar{u}(x) := \mathbb{E}_{\mathcal{O}}[u_{\mathcal{O}}(x)]. \quad (4.1)$$

There are two standard ways to approach this expectation, each of which loses one of the properties we want. The most direct is ensemble averaging, which repeatedly samples a particle configuration, solves the PDE on the sampled geometry, and averages the resulting solutions. This estimator is unbiased and useful as a baseline, but each solve requires a global sampling of the configuration. It therefore destroys the output sensitivity of estimators like WoS and quickly becomes intractable for large-scale media such as clouds.

Homogenization takes an alternative approach. Instead of preserving the particle model and averaging over realizations, it replaces the particles with a proxy PDE derived from their asymptotic behavior [Marchenko and Khruslov 2008; Papanicolaou 1995; Giunti et al. 2018]. For absorbing particles under the Laplace equation, this proxy is a screened equation posed without any explicit microparticle geometry. Homogenization therefore avoids global sampling and can be solved with standard elliptic PDE solvers, but it describes a limiting regime rather than the original particle model. Away from this regime, the estimates are biased, often non-uniformly across the domain in ways that depend on the local scale of geometric features and are difficult to assess or correct.

We instead take inspiration from volume rendering, where light transport in clouds, smoke, fog, and tissue is routinely simulated without sampling entire configurations or homogenizing the medium [Drebin et al. 1988; Raab et al. 2006; Novák et al. 2018]. There, mean radiance satisfies a recursive integral equation whose dependence on the medium enters through a free-flight distribution. Sampling this distribution gives a random distance to the next particle interaction, replacing the deterministic ray-intersection query used for fixed surfaces. This suggests a *volumetric* analogue of Monte Carlo PDE solvers: rather than querying an explicit boundary, the estimator samples local interaction events from the underlying particle model.

We develop this idea for elliptic PDEs on random microparticle geometry. As in preceding chapters, we begin from an integral equation, but now derive it for the mean solution $\bar{u}$ rather than the solution on a fixed domain. Using the Poisson Boolean model and its closest-point distribution as the underlying stochastic geometry primitive, we replace deterministic closest-boundary queries in WoS with stochastic closest-particle queries (Figure 4.3, left). These queries play the same role for WoS that free-flight sampling plays for volume rendering, letting the recursive estimator interact with the medium without constructing a complete particle configuration. Unlike ray-based volume rendering, however, closest-point queries during a walk must be conditioned on past samples from the medium; maintaining this partial realization is a key algorithmic development of the chapter.

At a high level, the mean solution has the following recursive integral representation,

Volumetric walk on spheres

$$\bar{u}(x) = \int_{\mathbb{R}^3} p_x(q) \int_{\partial B(x, \|q-x\|)} P(x, y) \bar{u}(y \mid q^* = q) dy dq. \quad (4.2)$$

The outer integral averages over the random closest point induced by the medium, while the inner integral is the usual mean-value relation over the resulting sphere. The conditional in $\bar{u}(y \mid q^* = q)$ reflects the fact that a closest-point sample reveals information about the medium,

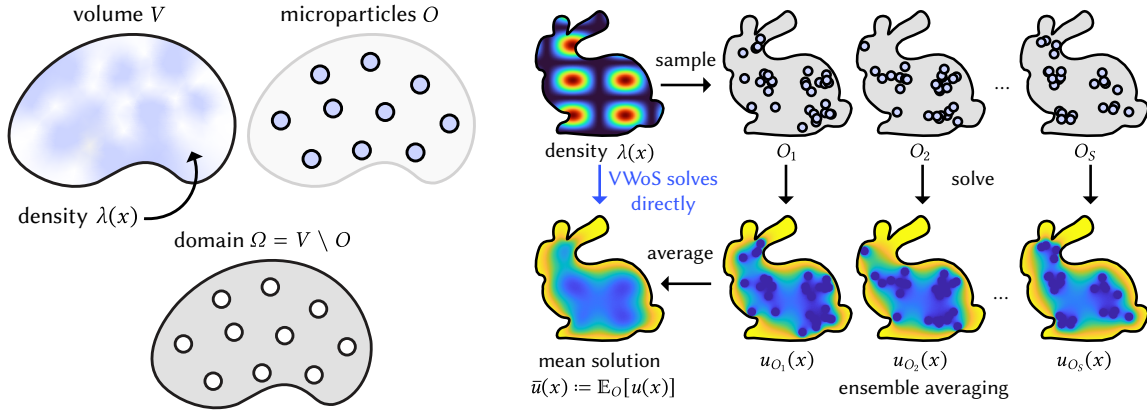


Figure 4.2: (Left) We consider domains $\Omega_O := V \setminus O$, equal to the difference between a deterministic volume V and a random configuration of particles O contained in the volume, drawn from a Poisson Boolean model whose density λ is defined over the volume. (Right) Ensemble averaging is a simple but expensive way to estimate the mean solution: sample many particle configurations (top), solve the PDE on each realized domain (bottom), and average. Volumetric walk on spheres instead estimates the mean solution directly.

and therefore changes the distribution over particle configurations for subsequent evaluations. Making the recursion precise requires a corresponding integral representation for this conditional mean solution, which we derive in Section 4.2.

Before deriving this estimator, we first formulate the class of boundary value problems induced by participating media. We then derive volumetric walk on spheres, including the conditional recursion needed to maintain consistency across recursive samples, and its extension to reflecting particles.

4.1 Boundary Value Problems in Participating Media

We first formalize the boundary value problems considered in this chapter. Let $V \subset \mathbb{R}^3$ denote the macroscopic volume occupied by the participating medium, and let $O \subset V$ denote the random microparticle geometry contained in this volume. For each realization of O , the PDE is posed on the region not occupied by particles (Figure 4.2),

$$\Omega_O := V \setminus O. \quad (4.3)$$

The boundary of this realized domain consists of the deterministic boundary ∂V of the volume together with the random particle boundary ∂O . Thus, a boundary value problem in a participating medium is a random family of deterministic boundary value problems, one for each possible particle configuration.

For clarity, we begin with the Laplace equation with Dirichlet boundary conditions. Given a particle configuration O , let u_O solve

$$\begin{aligned} \Delta u_O(x) &= 0, & x &\in \Omega_O, \\ u_O(x) &= g(x), & x &\in \partial\Omega_O. \end{aligned} \quad (4.4)$$

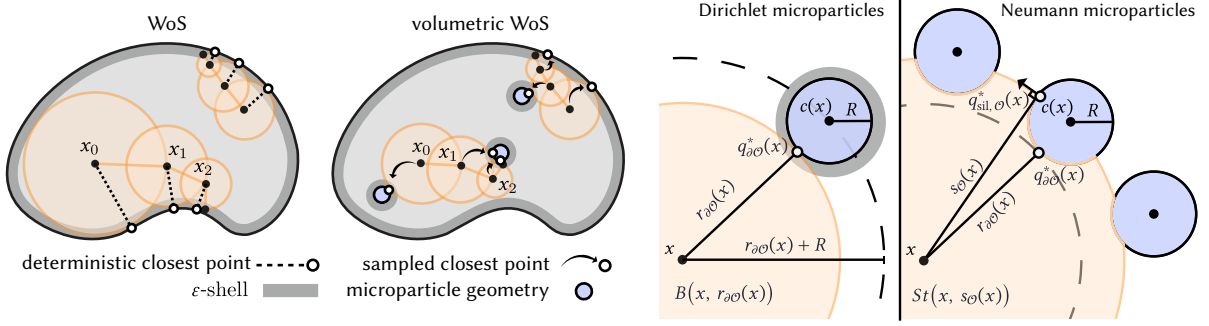


Figure 4.3: (Left) WoS jumps along spheres whose radii are set by deterministic closest-point queries; volumetric WoS jumps the same way, but samples closest points from the unresolved microparticle geometry, realizing only the particles each walk encounters. (Center right) With absorbing particles, sampling the closest center $c(x)$ determines both the closest point $q_{\partial O}^*(x)$ and its distance $r_{\partial O}(x)$ —the radius of the next empty ball. (Right) With reflecting particles, the star-shaped region St is bounded instead by the nearer of the deterministic boundary and the closest particle silhouette $q_{\text{sil}, O}^*(x)$.

Here g prescribes the boundary value on both the outer volume boundary and the particle boundaries. Since particles may appear throughout V , it is convenient to view g as a function defined over the volume, rather than only on a fixed boundary. This convention lets the same boundary data be evaluated on any realized particle surface.

Mean Solution over Particle Configurations. The solution u_O is random because the domain Ω_O is random. The quantity of interest is the mean solution, obtained by averaging over the space $\mathcal{P}(V)$ of possible particle configurations in the volume,

$$\bar{u}(x) = \int_{\mathcal{P}(V)} p(O) u_O(x) dO. \quad (4.5)$$

Here $p(O)$ and the measure dO are determined by the stochastic model of microparticle geometry. The next section specializes this model to the Poisson Boolean model, but the definition of the mean solution does not depend on that choice.

This distinction between u_O and $\bar{u}$ is the main conceptual difference from the fixed-domain problems considered earlier in the thesis. For any fixed O , one could apply an ordinary deterministic solver, including WoS, to estimate u_O . Repeating this process and averaging would recover $\bar{u}$ by ensemble averaging (Figure 4.2, right). The goal of this chapter is instead to derive estimators for $\bar{u}$ directly, so that the random particle geometry is sampled only through the local geometric events needed by the recursive solver.

4.2 Volumetric Walk on Spheres

We now derive the integral representation underlying volumetric walk on spheres (VWoS). The goal is to estimate the mean solution $\bar{u}$ without first sampling a complete particle configuration. The starting point is the usual WoS mean-value relation on each realized domain, but placed inside the expectation over particle configurations.

Expected Mean-Value Relation. For a fixed particle configuration $\mathcal{O}$, let

$$q_O^*(x) := \operatorname{argmin}_{y \in \partial\Omega_O} \|y - x\|, \quad r_O(x) := \|q_O^*(x) - x\| \quad (4.6)$$

denote the closest boundary point and its distance from x . The mean-value relation (2.10) holds on each realized domain Ω_O ; taking its expectation over particle configurations gives

$$\bar{u}(x) = \mathbb{E}_O \left[\int_{\partial B(x, r_O(x))} P(x, y) u_O(y) dy \right]. \quad (4.7)$$

This equation is the direct analogue of the deterministic WoS relation for a participating medium. The only difference is that the integration sphere is now random, because its radius is determined by the closest boundary point in the realized particle configuration.

Conditioning on the Closest Point. To separate this random geometry from the mean-value recursion, we condition on the closest point. Let $p_x(q)$ denote the probability density of $q_O^*(x)$. By the law of total expectation,

$$\bar{u}(x) = \int_{\mathbb{R}^3} p_x(q) \mathbb{E}_O \left[\int_{\partial B(x, \|q-x\|)} P(x, y) u_O(y) dy \mid q_O^*(x) = q \right] dq. \quad (4.8)$$

Conditioning on $q_O^*(x) = q$ fixes the sphere used by the mean-value relation. The remaining uncertainty is the solution induced by particle configurations consistent with this observed closest point. Moving the expectation inside the sphere integral gives

$$\bar{u}(x) = \int_{\mathbb{R}^3} p_x(q) \int_{\partial B(x, \|q-x\|)} P(x, y) \bar{u}(y \mid q_O^*(x) = q) dy dq. \quad (4.9)$$

Equation (4.9) has the form of a recursive mean-value relation, but it is not yet a closed recursion. The left-hand side is the unconditional mean solution, while the recursive term is conditioned on the closest point just sampled.

Memory and Conditional Recursion. After k steps, we write the accumulated conditioning events as the memory

$$M_k := (q_O^*(x_0) = q_0) \wedge \cdots \wedge (q_O^*(x_{k-1}) = q_{k-1}), \quad M_0 := \emptyset. \quad (4.10)$$

Thus, the recursive unknown is the conditional mean solution $\bar{u}(x_k \mid M_k)$. Applying the same conditioning argument at step k gives

$$\bar{u}(x_k \mid M_k) = \int_{\mathbb{R}^3} p_{x_k}(q_k \mid M_k) \int_{\partial B(x_k, \|q_k - x_k\|)} P(x_k, x_{k+1}) \bar{u}(x_{k+1} \mid M_k \wedge (q_O^*(x_k) = q_k)) dx_{k+1} dq_k. \quad (4.11)$$

This is the recursive integral equation estimated by volumetric walk on spheres. Compared to ordinary WoS, the closest boundary point is sampled from a closest-point distribution, conditioned on memory after the first step, rather than obtained from a deterministic closest-point query. The conditioning ensures that future samples remain consistent with the closest-point observations already made by the walk.

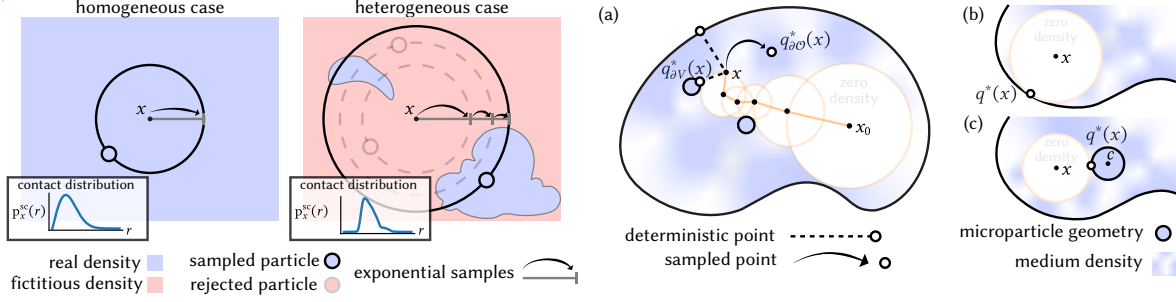


Figure 4.4: (Left) The unconditional closest-point sampler: in a homogeneous medium, an exponential variate determines the distance to the closest particle center; in a heterogeneous medium, thinning fills the volume with fictitious density and rejects candidates until a real particle is accepted. The sampled distance follows the contact distribution p_x^{dc} . (Right) Conditional sampling during a walk: (a) each step compares the sampled stochastic candidate $q_{\partial O}^*$ against the deterministic candidate $q_{\partial V}^*$ and previously realized geometry, taking the nearer point q^* ; (b, c) the memory of empty regions and sampled particles zeroes the density where the medium has already been observed, keeping later samples consistent with earlier ones.

The Volumetric Walk on Spheres Estimator. A single-sample Monte Carlo estimator follows by sampling the two integrals in Equation (4.11). At step k , sample

$$q_k \sim p_{x_k}(\cdot \mid M_k), \quad r_k := \|q_k - x_k\|, \quad (4.12)$$

then sample x_{k+1} on $\partial B(x_k, r_k)$ according to the normalized mean-value kernel. The memory is updated by appending the new closest-point event,

$$M_{k+1} := M_k \wedge (q_O^*(x_k) = q_k). \quad (4.13)$$

With this sampling convention, the estimator can be written in the same recursive form as ordinary WoS:

$$\widehat{u}(x_k \mid M_k) := \begin{cases} g(x_k), & r_k < \varepsilon, \\ \widehat{u}(x_{k+1} \mid M_{k+1}), & \text{otherwise.} \end{cases} \quad (4.14)$$

Equation (4.14) is the volumetric walk on spheres estimator. One edge case deserves mention: because particle geometry is realized lazily, a walk point can land *inside* a newly sampled particle, in which case the closest-point query returns a non-positive distance that we treat as $r_k = 0$. The termination criterion $r_k < \varepsilon$ absorbs this case, ending the walk just as it would in the ε -shell of resolved geometry. The remaining task is to adapt the Poisson Boolean closest-point sampler reviewed in the background chapter to the conditional distribution $p_{x_k}(\cdot \mid M_k)$. The next section describes this adaptation, including deterministic boundaries and the memory-dependent density used during a walk.

4.3 Conditional Closest-Point Sampling

The estimator in Equation (4.14) requires samples from the conditional closest-point distribution $p_x(\cdot \mid M)$. Our starting point is the unconditional Poisson Boolean closest-point sampler re-

viewed in Section 2.5 (Figure 4.4, left). Given a query point x and a particle density $\lambda : V \rightarrow \mathbb{R}_{\geq 0}$, this sampler returns a closest point $q_{\partial O}^*$ on the unresolved stochastic particle geometry, using thinning in the heterogeneous case. This section describes the two modifications needed for VWoS: comparing the stochastic sample against deterministic geometry, and conditioning the sampler on the memory accumulated by the walk. Throughout, we distinguish three closest points: $q_{\partial O}^*$, the candidate on the unresolved stochastic particles; $q_{\partial V}^*$, the closest point on the deterministic boundary; and q^* , the nearer of the two that the walk actually uses. These refine the realized-geometry closest point q_O^* of Section 4.2, which assumed the full particle configuration was known.

Deterministic and Stochastic Geometry. A boundary value problem in a participating medium contains both stochastic particle geometry and deterministic geometry. The deterministic part includes the boundary ∂V of the participating medium. Therefore, a closest-point sample for the full domain is obtained by comparing two candidates. First, we sample a closest point $q_{\partial O}^*$ on the stochastic particle geometry. Second, we compute the deterministic closest point

$$q_{\partial V}^* = \operatorname{argmin}_{y \in \partial V} \|y - x\|. \quad (4.15)$$

The closest point used by the walk is the nearer of these two candidates,

$$q^* = \operatorname{argmin}_{y \in \{q_{\partial O}^*, q_{\partial V}^*\}} \|y - x\|. \quad (4.16)$$

This is analogous to volume rendering with surfaces: one samples a possible medium interaction, also checks the nearest surface intersection, and uses whichever event occurs first. The discussion so far describes an unconditional closest-point sample: it accounts for both stochastic particles and fixed geometry, but not for information revealed by previous steps of the walk. During VWoS, each sampled closest point becomes part of the conditioning state for later samples. This modifies the same sampling procedure rather than introducing a new one.

Conditioning on Empty Regions. Conditioning on walk memory modifies the closest-point sampler in two ways. First, each previous closest-point sample certifies that a ball around the corresponding walk point is empty. Let $\mathcal{E}(M)$ denote the union of these empty balls. Future stochastic particles must be prevented from intersecting this region. For spherical particles of radius R , this is achieved by zeroing the particle-center density inside a dilation of the empty region:

$$\lambda_M(z) = \begin{cases} 0, & z \in \mathcal{E}(M) \oplus R, \\ \lambda(z), & \text{otherwise.} \end{cases} \quad (4.17)$$

Here $\mathcal{E}(M) \oplus R$ denotes dilation by a ball of radius R . Sampling unresolved particles with λ_M ensures that later samples do not contradict regions already known to be empty (Figure 4.4, right).

Conditioning on Sampled Particles. Second, some previous closest-point samples may reveal particles. Let $C(M)$ denote the union of particles sampled so far. These particles are no longer part of the unresolved stochastic medium; they are treated as deterministic geometry for the rest of the walk. Thus, under memory M , the deterministic closest-point query is performed against

$$\partial V \cup \partial C(M). \quad (4.18)$$

Conditional closest-point sampling combines these two modifications. We sample the stochastic closest point using the conditioned rate λ_M , query the closest point on $\partial V \cup \partial C(M)$, and return the nearer of the two. After this closest point is selected, the memory is updated by adding the new empty ball. If the selected point came from a newly sampled stochastic particle, that particle is also added to $C(M)$. If the selected point came from deterministic geometry or an already sampled particle, no new particle is added.

Memory-Dependent Closest-Point Queries. This procedure supplies the conditional distribution $p_x(\cdot | M)$ used in Equation (4.14). The role of memory is therefore not to introduce a new stochastic model, but to update two parts of the existing query procedure: the density used to sample unresolved particles, and the deterministic geometry used for closest-point queries. This conditioning issue also exists in volume rendering. A free-flight sample reveals empty space along a ray and may reveal a particle interaction, so later samples should in principle condition on that information.

In practice, volume rendering usually ignores this dependency because rays expose only thin regions of the medium, and the probability of later encountering the same sampled particle or empty region is negligible when particles are sufficiently small relative to the scene. For closest-point sampling, the effect is more significant: each sample constrains a ball around the walk point, which can strongly affect later closest-point queries (Figure 4.5, right). In our experiments, we store the resulting empty spheres and sampled particles in linear arrays. This representation is simple and sufficient for the examples in this chapter, but it can become inefficient for long walks because each query may need to check many previously sampled regions and particles.

4.4 Volumetric Walk on Stars

The same construction extends to reflecting particles by replacing WoS with WoSt; we call the result volumetric walk on stars (VWoSt). To keep the discussion simple, suppose the deterministic boundary of the medium carries Dirichlet data, while the stochastic particles are perfectly reflecting. For each particle configuration O , the realized problem is

$$\begin{aligned} \Delta u_O(x) &= 0, & x &\in \Omega_O, \\ u_O(x) &= g(x), & x &\in \partial V, \\ \frac{\partial u_O(x)}{\partial \vec{n}} &= 0, & x &\in \partial O. \end{aligned} \quad (4.19)$$

Thus the walk should terminate on the deterministic boundary ∂V , but reflect from particle boundaries.

Random Star-Shaped Domains. For a fixed configuration, WoSt replaces the largest empty sphere of WoS by a star-shaped region. Its radius is determined by the nearer of two events: the closest Dirichlet point on ∂V , or the closest visibility silhouette of the reflecting boundary ∂O . If $q_{\partial V}^*(x)$ is the deterministic closest point on ∂V and $q_{\text{sil},O}^*(x)$ is the closest particle silhouette point (Figure 4.3, right), then

$$s_O(x) := \min \left\{ \|q_{\partial V}^*(x) - x\|, \|q_{\text{sil},O}^*(x) - x\| \right\}. \quad (4.20)$$

Averaging the corresponding WoSt mean-value relation over particle configurations gives the same kind of recursive mean-solution equation as in volumetric WoS, except that the random geometric query is now a closest-silhouette query rather than a closest-point query:

$$\bar{u}(x) = \mathbb{E}_O \left[\int_{\partial St(x, s_O(x))} P(x, y) u_O(y) dy \right]. \quad (4.21)$$

The same conditioning and memory logic applies: once a silhouette or closest-point event is sampled, later geometric queries must remain consistent with that partial realization of the medium.

Conditional Silhouette Sampling. For spherical particles, closest-silhouette sampling can be built directly from the closest-point sampler already described. The closest silhouette of a sphere from x lies on the same particle as the closest point. Therefore, we first sample the closest particle using the conditional closest-point sampler, then compute the corresponding silhouette point analytically. The silhouette distance can be larger than the closest-point distance, so forming the full star-shaped region may require continuing to sample particles in increasing distance order until the sampler has passed the current silhouette radius. Any particles found before that radius are added to the walk memory as sampled deterministic geometry. In this way, volumetric WoSt uses the same stochastic particle model and the same memory mechanism as volumetric WoS, but replaces closest-point events by the visibility-silhouette events needed to construct star-shaped regions.

4.5 Evaluation

The experiments in this section validate that the volumetric estimators compute the intended mean solution, quantify when they improve over direct baselines, and illustrate how memory enables coupling with other Monte Carlo solvers.

Comparison to Ensemble Averaging. We first compare against ensemble averaging, which provides a direct reference for the mean solution by sampling complete particle configurations, solving the deterministic problem on each realization, and averaging the resulting estimates. Across both Dirichlet and mixed-boundary experiments, VWoS and VWoSt match these ensemble-averaged references up to Monte Carlo error, while avoiding the global particle sampling and geometric query costs of explicit configurations. For equal numbers of walks, VWoS is already

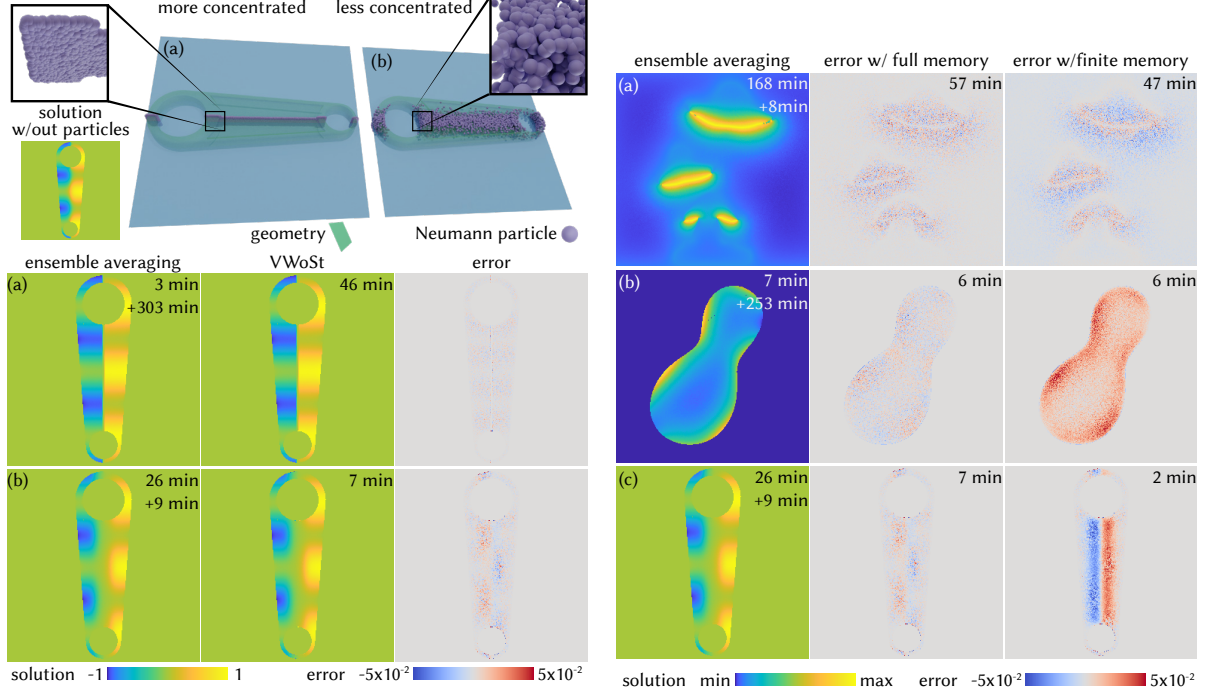


Figure 4.5: (Left) Mean solutions with reflecting particles: across more and less concentrated media, VVoSt matches ensemble-averaged references up to Monte Carlo error while being nearly $5\times$ faster. (Right) The need for memory: truncating the walk’s memory to finite size improves runtime marginally or not at all, yet always introduces significant bias, across both absorbing and reflecting particles.

more than $3\times$ faster than ensemble averaging, and VVoSt is nearly $5\times$ faster in the mixed-boundary experiments (Figure 4.5, left). The advantage becomes much larger for localized evaluation: in the membrane example, evaluating a 256×256 slice takes 11 seconds with ensemble averaging and 1 second with VVoS, while evaluating along a line is more than $10,000\times$ faster. This reflects the central advantage of the method: ensemble averaging must instantiate particles throughout the medium, whereas the volumetric estimators sample only the stochastic geometry encountered by each walk.

Comparison to Homogenization. We also compare against homogenization, which replaces the random particle geometry by an effective PDE derived from an asymptotic particle regime. Homogenization can be faster—in our experiments, roughly $2\text{--}20\times$ faster than VVoS—but this speed comes from solving a proxy problem rather than estimating the original mean solution. As particle radius increases, the homogenized solution develops visible bias; for fixed particle radius, the bias is larger near thin geometric features and near the medium boundary, where the particle scale is no longer negligible relative to the domain (Figure 4.6). VVoS instead samples the original stochastic particle model, so its error appears as Monte Carlo variance rather than bias from an asymptotic PDE. In the membrane example, this direct treatment of

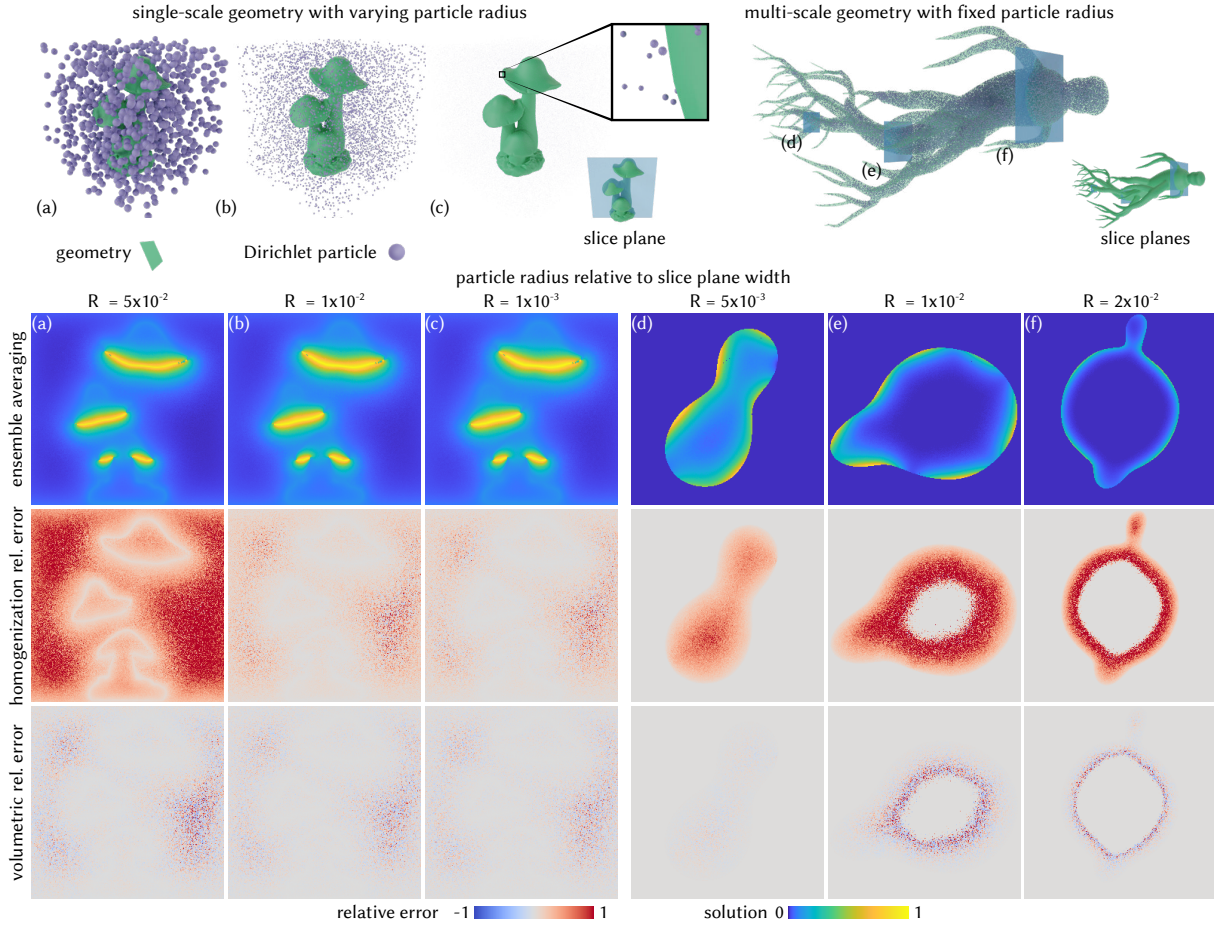


Figure 4.6: Compared to a reference mean solution from ensemble averaging, VVoS produces unbiased estimates across particle sizes, whereas homogenization develops visible bias as particle radius grows—largest near thin geometric features and the medium boundary, where the particle scale is no longer negligible.

particles adds only about 15% runtime over running WoS without particles, while capturing electrostatic screening effects that are absent from the particle-free solve.

Coupled Monte Carlo Simulation. Finally, we use VVoS in a coupled cloud photochemistry example to demonstrate the method on a larger participating-medium simulation (Figure 4.7). In this toy model, volumetric path tracing estimates the incident fluence from sunlight inside a cloud, while VVoS estimates diffusion of the resulting chemical species. The example highlights two aspects of the formulation. First, VVoS can be combined naturally with volume rendering because both estimators operate on the same Poisson Boolean medium. Second, the memory accumulated by the VVoS walk must be carried into the path tracer so that both estimates condition on the same partial realization of the medium. Conditioning both estimators consistently changes the resulting concentration field, capturing shadowing effects that are lost if the light-transport estimate is sampled independently. This example illustrates how stochastic geometric

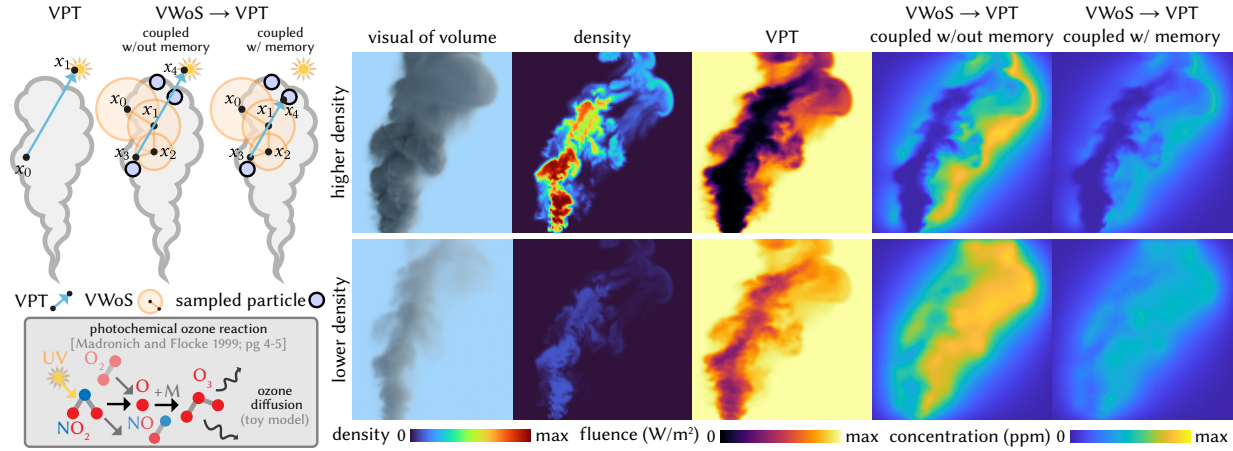


Figure 4.7: Coupled Monte Carlo simulation of cloud photochemistry: primary rays trace sunlight to its first collision inside the cloud, where the absorbed energy generates ozone, and VVoS estimates the diffusion of the resulting chemical species. Both estimators operate on the same Poisson Boolean medium, and sharing the walk’s memory keeps their estimates consistent, capturing shadowing effects lost under independent sampling.

queries can serve as a shared interface for coupled Monte Carlo simulation in participating media.

4.6 Outlook

Richer Stochastic Media. The estimator developed in this chapter uses one of the simplest stochastic models of participating media: independent spherical particles sampled from a Poisson Boolean model. This choice makes the closest-point distribution tractable, but it leaves much of the modeling space unexplored. Many physical media involve anisotropic particles, broad size distributions, or correlated particle positions. Volume rendering has already developed models and estimators for several of these cases, including anisotropic and non-exponential media [Jakob et al. 2010; Heitz et al. 2015; Bitterli et al. 2018; Jarabo et al. 2018; d’Eon 2018]. Extending VVoS and VVoSt to such models would broaden the class of microparticle systems that can be simulated without ensemble averaging or homogenization.

Improved Sampling Techniques. A second direction is to improve the sampling machinery. The implementation here uses the closest-point analogue of delta tracking: sample from a majorant process and reject candidates until a real particle is accepted. This is robust, but can become inefficient when the density is highly heterogeneous or concentrated. Rendering has developed many techniques for this regime, including residual and ratio tracking, next-event estimators, transmittance estimators, and multiple-importance sampling for null paths [Novák et al. 2014; Kutz et al. 2017; Miller et al. 2019; Georgiev et al. 2019; Kettunen et al. 2021]. Adapting these ideas to closest-point and closest-silhouette sampling is a natural way to reduce variance and improve performance.

Geometric Acceleration Structures. Finally, the geometric data structures used by the estimator remain deliberately simple. Our implementation repeatedly compares stochastic samples against deterministic geometry and stores walk memory as linear arrays of empty spheres and sampled particles. This is adequate for the examples in this chapter, but it is not optimal for long walks or dense media. Future implementations could use conservative distance bounds, or acceleration data structures for memory, reducing both repeated deterministic closest-point queries and linear scans through previously sampled regions. These improvements would not change the estimator, but they could make conditional closest-point sampling substantially more practical.

Chapter 5

Caching-Based Acceleration

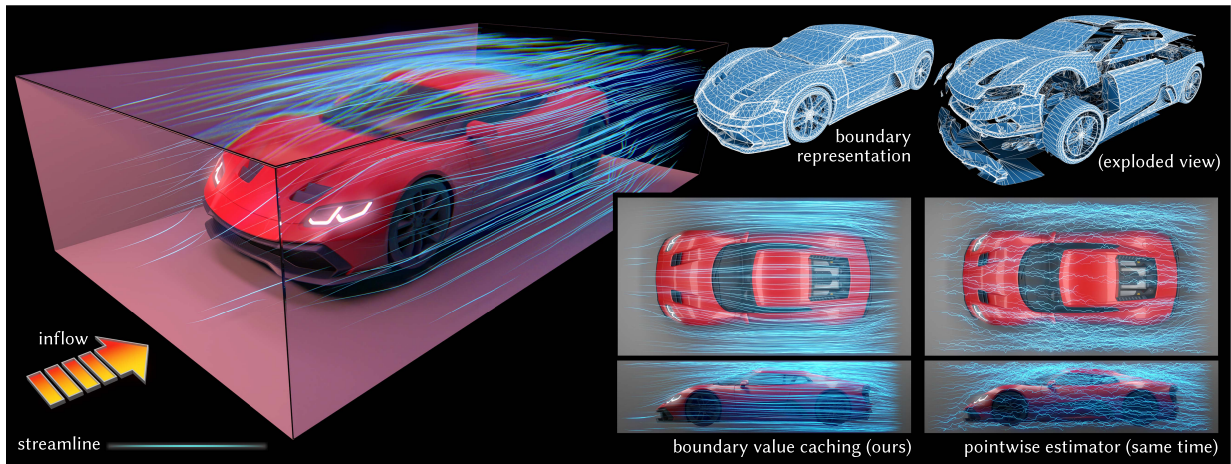


Figure 5.1: Streamlines traced through a simulated wind tunnel. Because the gradient comes from the same cached boundary samples as the solution, a single cached solve traces the entire flow; at equal time, cached streamlines are smoother and better converged than pointwise estimation (Section 5.3).

Recasting linear elliptic PDEs as recursive integral equations yields Monte Carlo estimators that inherit much of rendering’s geometric scalability as well as its convergence machinery. Like their counterparts in rendering, these estimators are efficient pointwise and scale comfortably to tens of thousands of evaluation points. But a dense solution over a large domain exposes a different inefficiency: solving every point independently is wasteful. A 1024^2 image costs a million times more than a single pixel, a 1024^3 grid a billion times—and much of that cost is redundant, repeated between nearby points whose walks pass through nearly the same geometry and carry nearly the same information.

Rendering long ago addressed the same redundancy through *caching* methods. Rather than tracing full paths from every pixel, methods like virtual point lights [Keller 1997; Dachsbacher et al. 2014] and photon mapping [Jensen 1996; Hachisuka and Jensen 2009] deposit radiance throughout the scene once, then perform a single cheap *final gather* from each pixel to query it. This decouples the resolution of the image from the cost of simulating transport. The same opportunity exists for Monte Carlo PDE solvers—and, we argue, a greater one, because the cached

quantity is more amenable to reuse in three respects. Radiance is five-dimensional, varying across position and direction; a PDE solution is purely spatial and lower-dimensional, so it is cheaper to cache and query. Radiance can be a high-frequency signal, whereas solutions to elliptic PDEs are smooth by construction, so a single sample stays accurate over a wider neighborhood. And rendering queries radiance only on the image plane, whereas PDE applications often require the solution densely throughout the interior, so each cached sample reaches far more of the output. Taken together, these characteristics make Monte Carlo PDE solvers an even better fit for sample reuse than rendering, where caching has long been one of the most effective ways to amortize the cost of simulating transport.

We develop this idea through *boundary value caching* (BVC). We estimate the unknown potential at sample points on the boundary, cache it, and splat each cached sample into the interior through the boundary integral equation [Costabel 1987; Hunter and Pullan 2001]—the direct analogue of depositing a virtual point light and gathering from it. Three properties make this the natural place to cache:

- reuse is *global and unrestricted*—a single boundary walk contributes to every interior evaluation point at once. A virtual point light reaches only the surfaces it can see, its contribution gated by a visibility term; the boundary integral equation carries no such term, so a cached boundary sample reaches the entire interior without occlusion;
- the reconstruction is *harmonic by construction*: not smooth merely in expectation, but exactly, for every realization of the cache—a property volumetric reuse cannot offer, and one we develop in Section 5.1; and
- *storage scales with a bounding surface rather than the enclosed volume*, the same dimensionality reduction that makes boundary methods attractive in the first place.

These properties come with practical considerations that must be addressed. Sharing samples across evaluation points forgoes importance sampling the singular kernels (Section 5.2), and the splat is an $O(NM)$ operation over N boundary samples and M evaluation points (Section 5.2).

Boundary value caching was introduced alongside two concurrent schemes that instead reuse samples throughout the volume, reverse walks [Qi et al. 2022] and mean value caching [Bakboug and Peers 2023]; we defer a discussion of surface against volumetric caching to Section 5.4, and focus here on the boundary integral formulation as the most direct analogue to rendering. We introduce the global boundary integral equation and the Monte Carlo estimator it induces (Section 5.1), develop the reuse of cached boundary data and the smoothness it confers, and address the practical cost of singular kernels and splatting (Section 5.2).

5.1 Boundary Integral Equation and Sample Reuse

We again consider the Laplace equation with Robin boundary conditions,

$$\Delta u(x) = 0 \quad \text{on } \Omega, \quad \frac{\partial u(x)}{\partial \vec{n}} - \mu u(x) = h(x) \quad \text{on } \partial\Omega. \quad (5.1)$$

While the construction extends to mixed boundary conditions and Poisson equations, we present it for the Robin problem of Chapter 3. For $x \in \Omega$, the boundary integral equation follows from

the regrouped identity (3.4), now applied over the boundary of the full domain:

$$u(x) = \int_{\partial\Omega} [P(x, y) - \mu G(x, y)] u(y) - G(x, y) h(y) dy. \quad (5.2)$$

This is the same representation that underlies the boundary element method [Costabel 1987; Hunter and Pullan 2001], rewritten to substitute Robin boundary data directly.

Sample Reuse on the Boundary. There are three important observations here. First, as an integral equation we can use a Monte Carlo estimator to evaluate this integral directly. Second, since $u(y)$ is the solution to a linear elliptic PDE, we can directly employ WoSt to estimate it rather than rely on a dense global solve of a linear system as in BEM [Hunter and Pullan 2001], which is moreover sensitive to the quality of the boundary triangulation (Figure 1.3, right). Finally, since this integral is shared across all points $x \in \Omega$ with variation only in the kernels as a function of evaluation point x , but not in the boundary data, we can reuse sample estimates across all evaluation points. We refer to this shared Monte Carlo estimation scheme as *boundary value caching* (BVC). For N uniform area samples on the boundary, the resulting estimator for any point in the domain is

Boundary value caching

$$\hat{u}(x) = \frac{|\partial\Omega|}{N} \sum_{i=1}^N ([P(x, y_i) - \mu G(x, y_i)] \hat{u}(y_i) - G(x, y_i) h(y_i)). \quad (5.3)$$

Estimating Unknown Boundary Data. The unknown boundary data $\hat{u}$ is estimated using the pointwise WoSt algorithm of Chapter 3. In the case of mixed-boundary problems, we simply solve for the complementary boundary data. For instance, on a part of the boundary with Dirichlet conditions we can use $u(x) = g(x)$ directly but have to estimate the unknown normal derivative $\frac{\partial u(x)}{\partial \vec{n}_x}$ in order to infer the unknown Robin data $h(x) = \frac{\partial u(x)}{\partial \vec{n}_x} - \mu g(x)$. Recent work [Yu et al. 2025] introduces unbiased estimators for the normal derivative directly on the boundary, but BVC originally evaluates it at a point offset into the domain, where the gradient estimator of Sawhney and Crane [2020] applies. For this construction to be consistent, the offset must exceed the ε -shell (in practice 5ε): a ball contained entirely within the shell terminates its walks immediately, driving the normal-derivative estimate toward zero and biasing the reconstructed interior solution.

Harmonic Reconstruction. Beyond enabling reuse, the BVC estimator inherits a structural convenience from the boundary integral equation. Since the boundary integral has no visibility term, a cached sample contributes to every evaluation point with no occlusion test. This global sharing is more than a runtime convenience—it makes the reconstruction *harmonic by construction*, a property the volumetric reuse schemes of bidirectional walk on spheres [Qi et al. 2022] and mean value caching [Bakbouk and Peers 2023] cannot offer. Because every interior point reads the same fixed set of boundary samples, the estimate is a superposition of monopoles

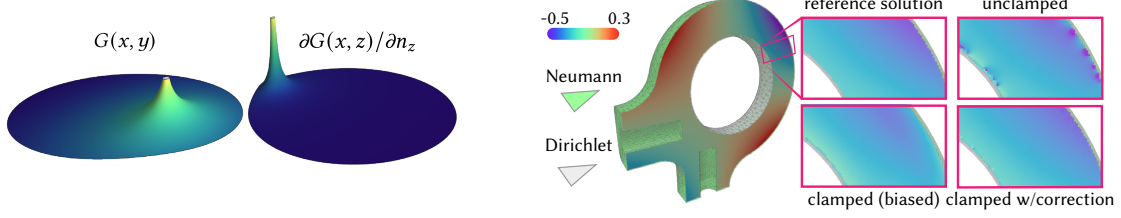


Figure 5.2: (Left) The free-space Green’s function and its normal derivative are singular at the evaluation point but decay smoothly and fall off quickly away from it. (Right) Because cached samples are shared rather than importance sampled, evaluation points near the boundary exhibit singular artifacts (unclamped); clamping the kernels suppresses the artifacts but biases the solution, while the correction step removes them without bias.

and dipoles, each harmonic in x away from its sample. Specifically, the Laplacian passes through the sum and distributes over the kernels,

$$\Delta_x \hat{u}(x) = \frac{|\partial\Omega|}{N} \sum_{i=1}^N (\Delta_x [P(x, y_i) - \mu G(x, y_i)] \hat{u}(y_i) - \Delta_x G(x, y_i) h(y_i)) = 0, \quad (5.4)$$

so the reconstruction is harmonic for every realization of the cache, not merely in expectation. Because the error against the true solution is itself harmonic, BVC exhibits a smooth, global discrepancy that vanishes as boundary samples accumulate, rather than the salt-and-pepper noise of independent pointwise estimates. The same design philosophy appears in recent stochastic barycentric coordinates [de Goes and Desbrun 2024], which preserve a structural property by construction within a Monte Carlo estimator. There the property is linear precision rather than harmonicity. Boundary value caching enforces the latter but not the former.

5.2 Practical Considerations

The properties that make boundary value caching attractive—global reuse, harmonicity, surface-scale storage—each carry a practical cost that a working implementation must address.

Kernel Singularities. Sharing samples across evaluation points forgoes importance sampling the kernels, so the singularity of $G(x, y)$ and $P(x, y)$ as $y \rightarrow x$ remains an issue for evaluation points near the boundary (Figure 5.2). The original BVC method, following the clamping strategy of instant radiosity and related many-light methods [Keller 1997; Dachsbacher et al. 2014], bounds the kernels and applies the bias-correction technique of Kollig and Keller [2006]—importance sampling the clamped residual with fresh walks—wherever the clamp is active. Recent work using the same kernels for surface reconstruction [Chen et al. 2024a] suggests a simpler and now more standard route—regularizing the kernel by convolution against a mollifier, which introduces a small, controllable bias but removes the singular artifacts outright.

Fast Summation. Beyond accuracy near the boundary, the dominant practical concern at scale is cost. Splatting a cached sample is far cheaper than the walk that produced it, so caching is a large net saving over running walks independently. At scale, however, the splatting itself becomes the bottleneck: contributing N boundary samples to M evaluation points is an $O(NM)$ operation. The original work does not address this, in part because even a naive linear scan already improves so dramatically on pointwise walk on spheres. But a mature body of work on fast summation removes exactly this kind of bottleneck. Barnes–Hut methods [Barnes and Hut 1986] aggregate the N source points for efficient far-field evaluation, reducing each query to $O(\log N)$ and the full splat over M evaluation points to $O(M \log N)$; fast multipole methods [Beatson et al. 1997] additionally aggregate query points for a further speedup. Both have been applied to the same kernels in graphics, for fast winding numbers [Barill et al. 2018] and fast dipole sums [Chen et al. 2024a]. Applying this approach here would introduce a small, controllable bias but is a natural way to remove the bottleneck.

Storage and Streaming. Storage admits two regimes. Caching on a fixed global set of boundary points and splatting from it costs $O(|\partial\Omega|)$ storage—still proportional to a bounding surface rather than the enclosed volume, the dimensionality reduction that motivated caching on the boundary in the first place. Alternatively, the sum in Equation (5.3) can be streamed: boundary samples are drawn in batches, splatted to every evaluation point, and discarded, with each point accumulating contributions in place. This requires no explicit cache at all, holding the storage overhead over the underlying pointwise estimator to $O(1)$.

Localized Reuse. Finally, reuse need not span the entire domain—a useful property for large scenes or for evaluation points clustered in an isolated region. Because the boundary integral equation of Equation (5.2) holds over any closed region, we can cache on the boundary ∂A of a subdomain $A \subset \Omega$ and reconstruct the solution inside A alone—useful when evaluation points are localized to a region of interest rather than spread across Ω . The one difference is that ∂A is a *virtual* boundary carrying no prescribed conditions, so the boundary data we substituted from the Robin condition is no longer available. We must instead estimate *both* unknowns on ∂A , the solution u and its normal derivative $\partial u / \partial \vec{n}$, and recover the solution from the general boundary integral equation

$$u(x) = \int_{\partial A} P(x, y) u(y) - G(x, y) \frac{\partial u(y)}{\partial \vec{n}_y} dy, \quad (5.5)$$

$$\hat{u}(x) = \frac{|\partial A|}{N} \sum_{i=1}^N \left(P(x, y_i) \hat{u}(y_i) - G(x, y_i) \frac{\partial \hat{u}(y_i)}{\partial \vec{n}_{y_i}} \right), \quad (5.6)$$

with both boundary quantities supplied by a shared WoSt evaluation. How best to place a virtual boundary, or to allocate samples across it given the distribution of evaluation points, is a question we leave open.

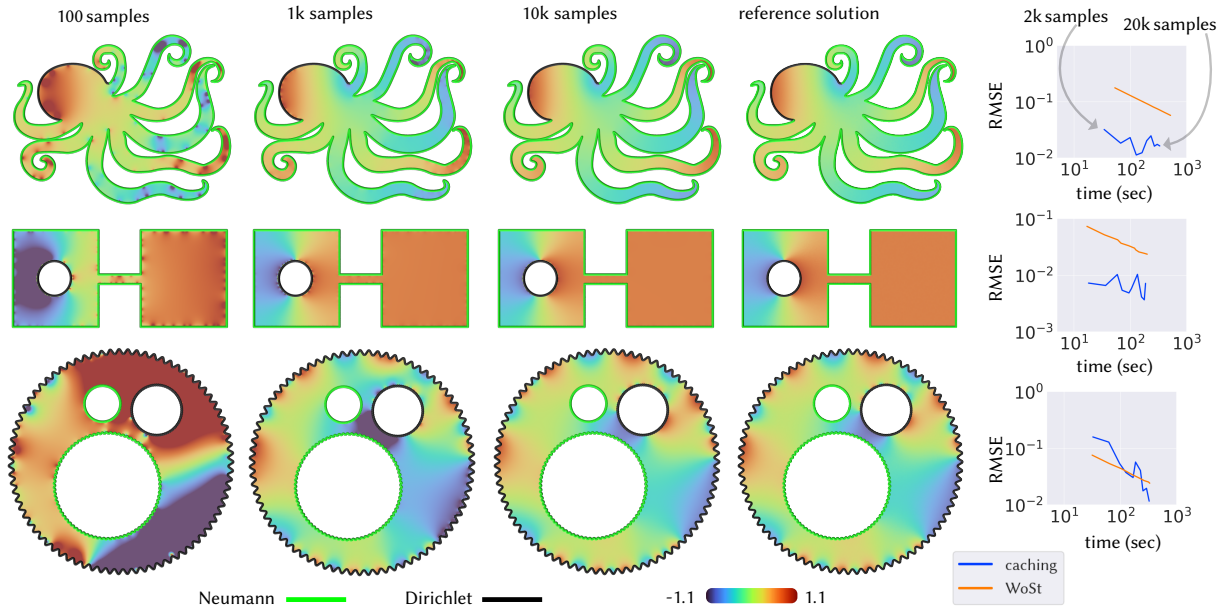


Figure 5.3: Cached reconstructions exhibit correlated rather than pointwise error: with few boundary samples the estimated solution is smooth but globally inaccurate, and this coherent global error vanishes as boundary samples accumulate.

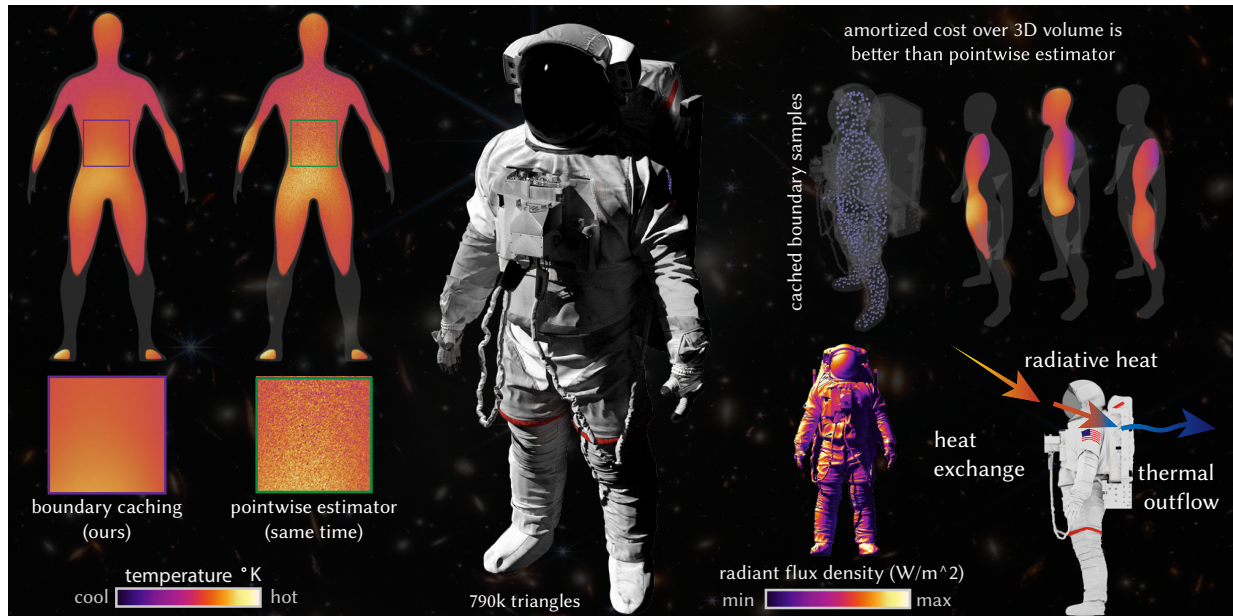


Figure 5.4: Thermal regulation of a spacesuit, with ray-traced solar radiation supplying Dirichlet data, outflow modeled by Neumann conditions, and an interior source for the astronaut's body heat. At equal time, the cached solution on a dense slice is markedly smoother than pointwise estimation.

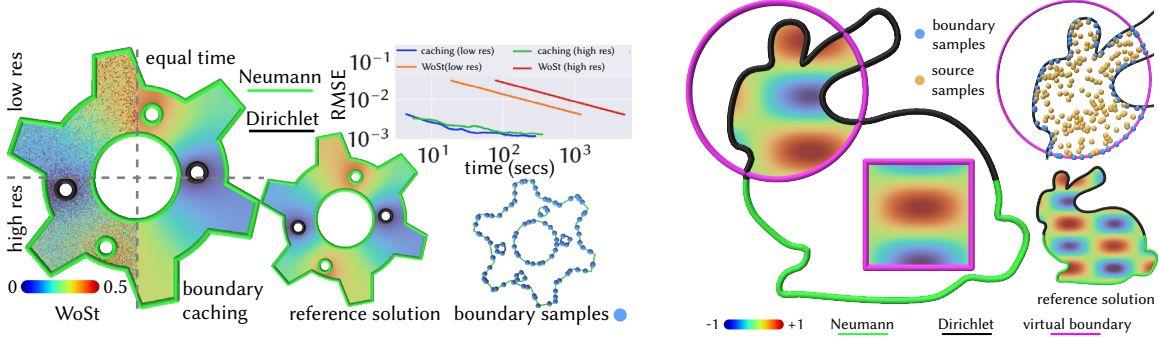


Figure 5.5: (Left) Caching yields far smoother results across the domain than pointwise estimation at equal time, and evaluating the boundary integral at more interior points has little impact on performance, since the same boundary samples determine the solution everywhere. (Right) Unlike a global solve involving the entire boundary, computation can focus on a region of interest by caching points only on the boundary of a smaller subdomain.

5.3 Evaluation

Dense Evaluation and Streamlines. We evaluate BVC on problems demanding the solution densely across a region, where reuse pays off, and find that it produces markedly smoother results than pointwise estimation at equal cost (Figure 5.4). Streamline tracing through a flow field is one such case: many points must be evaluated, and the result must stay smooth both across neighboring streamlines and along each one. Because the gradient comes from the same cached samples as the solution—differentiating the boundary integral equation simply moves ∇_x onto the kernels—a single cached boundary solve traces the entire flow, with no grid and no per-point walks. Against a pointwise solver at equal time, the cached streamlines are smoother and better converged (Figure 5.1).

Correlated Error. The smoothness this confers behaves unlike ordinary Monte Carlo error. With few samples the reconstruction is not noisy in the usual salt-and-pepper sense; it is smooth but globally inaccurate, and the error corrects itself as samples are added (Figure 5.3). Because every interior point reads the same cached samples, their estimates are correlated rather than independent, trading pointwise noise for a coherent global error that vanishes with more boundary data.

Localized Reuse. Finally, reuse can also be confined to a subdomain, reconstructing the solution in a localized region of interest at proportionally lower cost (Figure 5.5).

5.4 Outlook

Boundary versus Volumetric Caching. Boundary value caching has since been joined by a growing collection of schemes that instead cache throughout the volume [Qi et al. 2022;

Bakbouk and Peers 2023; Czekanski et al. 2024; Zhou et al. 2025; Huang et al. 2026]. Much of this drift toward volumetric caching traces to limitations of the original boundary formulation rather than to any intrinsic advantage—BVC as first presented lacked regularized kernels for the boundary singularity and fast-summation acceleration for the $O(NM)$ splat, the very gaps identified in Section 5.2. When these are addressed, the case for caching on the volume weakens considerably: it forgoes the global reuse and harmonicity that the boundary formulation provides by construction, and reintroduces storage proportional to the volume—precisely the dense grid these grid-free methods should ideally escape. While neural caching does not face the same storage overhead as other volumetric caching methods [Li et al. 2023], it could still benefit from a boundary formulation. Fitting a neural representation to WoSt estimates of the boundary data is a lower-dimensional, and so easier, learning problem, and splatting these learned values to the interior with free-space kernels would yield a representation that is harmonic by construction on the domain interior.

Path-Space Estimators. Caching, surface or volume, is only one axis of variance reduction. Over the past three decades, rendering has continued to advance along the others—quasi-Monte Carlo [Keller 1996], next-event estimation [Coveyou et al. 1967], path guiding [Vorba et al. 2014], neural denoising [Zhang et al. 2024], precomputed transfer [Zhao et al. 2013]—with each only beginning to cross over to walk on spheres [Ho and Owen 2026b,a; Huang et al. 2025; Hsu et al. 2026; Jambon et al. 2026]. Underlying and potentially unifying many of these is a notion absent from the present setting: a path space. Since Veach [1998], rendering has recast its recursive transport equation as a single integral over complete light paths, which has served as the mathematical foundation for multiple importance sampling, Markov chain Monte Carlo, and bidirectional estimators. The estimators of Chapter 3 instead remain strictly recursive, and bidirectional walk on spheres [Qi et al. 2022] shows that reasoning more globally about path structure, while possible, is considerably more involved here than in Monte Carlo rendering. A path space would supply the mathematical foundation such schemes currently lack, but the obstacle to one is structural: rendering’s transport equation is Fredholm, integrating over a single fixed domain, whereas Volterra equations integrate over subdomains that shift at every step, leaving no natural shared domain to build a path space from.

Chapter 6

Differentiable Simulation

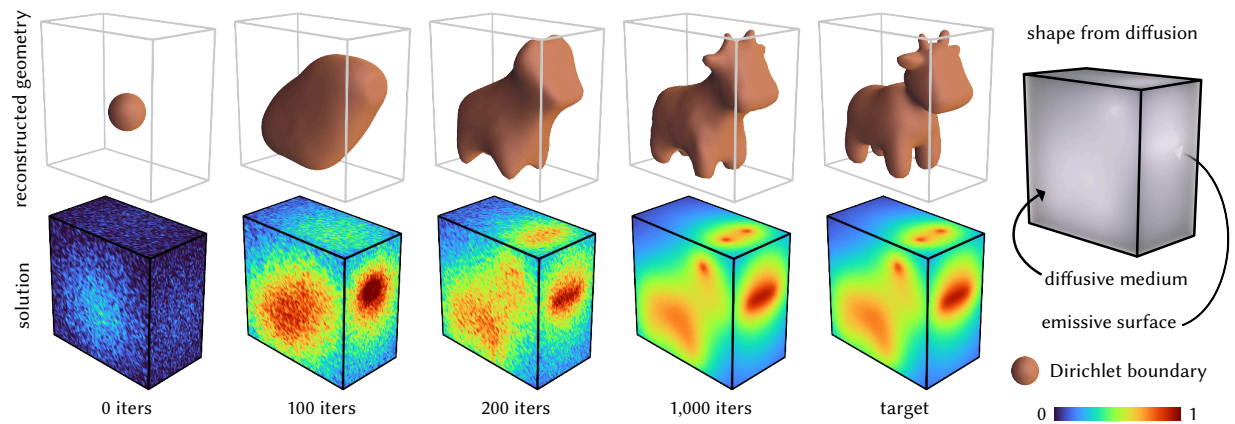


Figure 6.1: Differential walk on spheres recovers the shape of an emissive object from its diffusion profile, observed on a box surrounding the object, via gradient-based optimization. Unlike mesh- or grid-based approaches, derivatives are evaluated only at the points of interest—here, the observation locations on the box—without ever computing a global solution (Section 6.4).

A PDE solver estimates the *primal* solution—the function satisfying the governing equation and boundary conditions on a fixed geometry. We now turn to a different set of questions. Which geometry best explains an observed solution, and which geometry shapes the solution toward a desired one? Such inverse and design problems are fundamental across science and engineering. One might infer the shape of a tumor from a diffusion profile measured at the skin [Arridge 1999], or design a heat sink that minimizes the temperature at critical points of a circuit [Zhan et al. 2008]. Solving any of these problems requires differentiating the PDE solution with respect to the boundary geometry—the *differential quantities* that are the subject of this chapter [Henrot and Pierre 2018]. Computing differentials is a classical topic in numerical analysis; here we develop Monte Carlo estimators that inherit the same advantages as the primal solvers of the previous chapters: discretization-free evaluation, geometric flexibility, output sensitivity, and trivial parallelism.

A tempting route is to apply automatic differentiation directly to the primal solver [Griewank and Walther 2008; Baydin et al. 2018]. This is appealing for its simplicity but fails on two counts.

First, it is prohibitively expensive: a single walk takes many steps, and naively differentiating through them incurs exponential memory and quadratic compute, against the linear cost of the recursive forward algorithm [Nimier-David et al. 2020; Vicini et al. 2021]. Second, it mishandles the geometry. Differentiating a walk yields derivatives of the accelerated geometric queries it relies on—closest-point, ray intersection, visibility silhouette—which is both impractical [Niemeyer et al. 2020] and cannot correctly account for the evolving geometry that shape optimization requires.

Differentiable rendering encountered the same obstacle (Section 2.6). Directly differentiating a rendering algorithm can require storing large amounts of intermediate state, and a fixed differentiated execution trace does not by itself account correctly for derivatives induced by changing geometry, visibility, and boundary interactions. A successful line of work therefore avoids differentiating the implementation directly, and instead derives Monte Carlo estimators for the derivatives themselves [Gkioulekas et al. 2013; Li et al. 2018; Zhang et al. 2019; Loubet et al. 2019; Zhang et al. 2020]. These estimators closely parallel the corresponding primal rendering algorithms, and therefore retain many of their computational advantages. In modern systems, the primal quantity and its derivatives can even be estimated by a single solver [Jakob et al. 2022].

We take the same approach for boundary value problems. Rather than differentiating a Monte Carlo PDE solver, we formulate the boundary value problem satisfied by the solution derivative with respect to a set of parameters π . Classical results from shape optimization show that this derivative satisfies the same class of linear elliptic PDEs studied in the previous chapters, but with a boundary condition that depends on the primal solution [Henrot and Pierre 2018]—the elliptic analogue of the new emission term that drives differential radiance (2.28). Thus the recursive estimators developed earlier apply almost directly: the only new ingredient is that, upon reaching the boundary, the differential estimator must recursively estimate the primal solution appearing in the differential boundary condition.

This structure is well suited to gradient-based shape optimization. A single collection of nested walks estimates derivatives with respect to all parameters simultaneously, making it practical to optimize systems with thousands of degrees of freedom (Figure 6.6, right). At the same time, the estimator inherits the discretization-free nature of the primal walk-on-spheres solver, avoiding many of the robustness issues that can arise in mesh-based methods on evolving geometry. Finally, the resulting gradients are noisy but unbiased, making them naturally compatible with stochastic optimization. Early in optimization, inexpensive low-sample estimates are often sufficient, while the sample count can be increased as the iterate approaches a solution (Figure 6.6, left). As observed in differentiable rendering, stochasticity can even be beneficial, helping optimization avoid poor local minima [Gkioulekas et al. 2013; Li et al. 2018; Loubet et al. 2019; Zhang et al. 2020].

The core of the chapter develops this construction in detail. Section 6.1 derives the differential boundary value problem, Section 6.2 extends it to first-order linear boundary conditions, and Section 6.3 constructs the corresponding differential walk-on-spheres estimator.

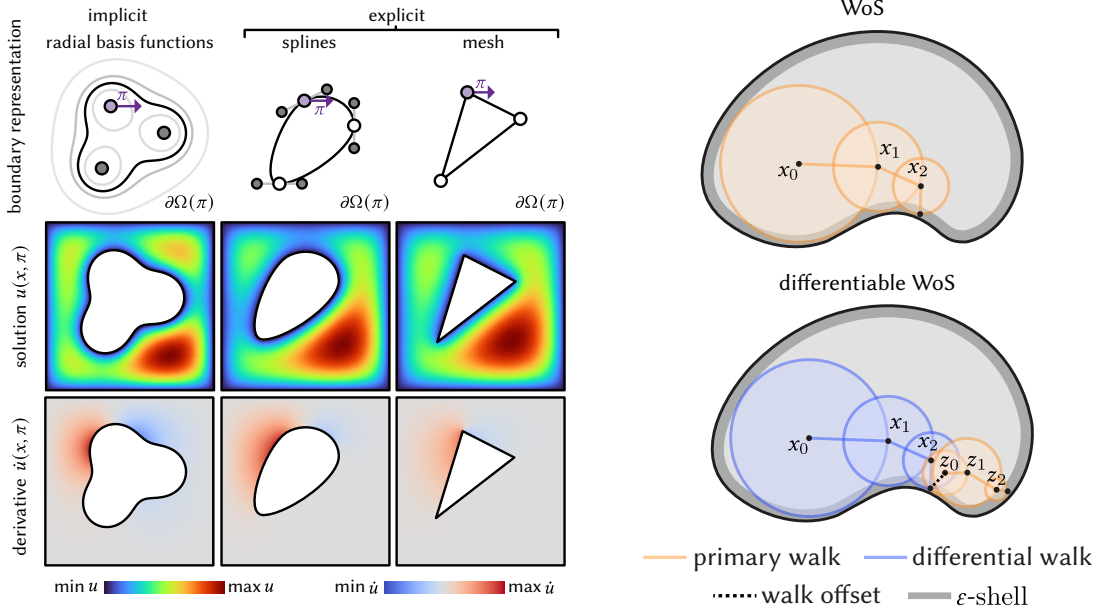


Figure 6.2: (Left) Derivatives can be computed on a wide range of boundary representations, including those not directly handled by conventional solvers—implicit radial basis functions, splines, and meshes. For a solution $u(x, \pi)$ (middle row), the derivative $\dot{u}(x, \pi)$ (bottom row) is taken with respect to positional parameters π of the boundary $\partial\Omega(\pi)$. (Right) Differential walk on spheres computes these derivatives. Like ordinary walk on spheres (WoS), the differential walk recursively jumps to a random point on the largest empty sphere until it terminates in the ϵ -shell—but there, instead of evaluating prescribed boundary data, it launches a primary walk from an offset point to estimate the differential boundary condition.

6.1 Differential Boundary Value Problems

We begin by formulating the shape derivative that will be estimated throughout this chapter. Let $\Omega(\pi) \subset \mathbb{R}^d$ be a domain controlled by parameters $\pi \in \mathbb{R}^M$, and let $\partial\Omega(\pi)$ denote its boundary. The parameters may encode, for example, mesh vertices, spline handles, radial basis function centers, or the pose of a rigid object (Figure 6.2, left). The tumor-imaging and thermal-design examples from the introduction are both instances of this objective: the former compares the solution to measurements on an observation region [Arridge 1999], while the latter penalizes high temperatures on design-critical components [Zhan et al. 2008]. We capture both cases using a shape functional

$$\mathcal{L}(\pi) = \int_{\Omega(\pi)} W(x) L(x, u(x, \pi)) dx, \quad (6.1)$$

where L measures the solution against an objective, and $W : \mathbb{R}^d \rightarrow [0, 1]$ localizes the objective to the region where measurements or design constraints are imposed. For example, L may be the squared error $\frac{1}{2}\|u - u_{\text{ref}}\|^2$ against a reference solution.

Differentiating Equation (6.1) using the Reynolds transport theorem gives [Henrot and Pierre 2018]

$$\frac{d\mathcal{L}(\pi)}{d\pi} = \int_{\Omega} W(x) L'(x, u(x, \pi)) \dot{u}(x) dx + \int_{\partial\Omega} W(x) L(x, u(x, \pi)) V_{\vec{n}}(x, \pi) dx, \quad (6.2)$$

where $\dot{u} := \partial u / \partial \pi$ is the solution derivative, and $V_{\vec{n}} = \vec{n} \cdot \partial x / \partial \pi$ is the normal velocity of the boundary under the parameter variation. The second term accounts for the motion of the integration domain, exactly as the edge integral of Equation (2.29) accounts for the motion of visibility discontinuities. It vanishes when the objective is supported away from the moving boundary, or more generally wherever the boundary contribution is zero. Equation (6.2) itself is valid only when the integrand of $\mathcal{L}$ carries no π -dependent discontinuities—satisfied here because the mask W is independent of π and L and u vary smoothly with the parameters. The central computational task is to estimate the solution derivative $\dot{u}$ in (6.2). We now derive the boundary value problem satisfied by $\dot{u}$. Suppose the primal solution solves the screened Poisson problem

Primal boundary value problem

$$\begin{aligned} \Delta u(x, \pi) - \sigma u(x, \pi) &= f(x) && \text{in } \Omega(\pi), \\ u(x, \pi) &= g(x, \pi) && \text{on } \partial\Omega(\pi). \end{aligned} \quad (6.3)$$

We assume here that the Laplacian, screening coefficient, and source are independent of π . Differentiating Equation (6.3) then yields a homogeneous screened Poisson equation for the derivative,

Differential boundary value problem

$$\begin{aligned} \Delta \dot{u}(x, \pi) - \sigma \dot{u}(x, \pi) &= 0 && \text{in } \Omega(\pi), \\ \dot{u}(x, \pi) &= D(x, \pi) - V_{\vec{n}}(x, \pi) \frac{\partial u}{\partial \vec{n}}(x, \pi) && \text{on } \partial\Omega(\pi). \end{aligned} \quad (6.4)$$

The boundary term follows by differentiating the identity $u(x(\pi), \pi) = g(x(\pi), \pi)$ at a moving boundary point $x(\pi) \in \partial\Omega(\pi)$. The chain rule gives a contribution from the parameter derivative of u and a contribution from moving the boundary through the normal derivative of the primal solution, producing the term $-V_{\vec{n}} \partial u / \partial \vec{n}$ [Henrot and Pierre 2018].

The differential problem has the same elliptic form as the primal problem, but with different source and boundary data. The important difference is that the boundary condition for $\dot{u}$ depends on the primal solution through its normal derivative. The two problems are therefore nested: when a differential walk reaches the boundary, evaluating its boundary condition requires estimating a primal quantity at that same boundary point. The term D collects the explicit parameter dependence of the prescribed Dirichlet data. Its form depends on how the boundary data is represented.

Restricted Boundary Data. The simplest case is when the Dirichlet values are obtained by restricting an ambient field $g_{\mathbb{R}^d}(\cdot, \pi)$ to the moving boundary. That is, a boundary point

$x \in \partial\Omega(\pi)$ receives the value $g_{\mathbb{R}^d}(x, \pi)$. Differentiating this value with respect to π is just the chain rule: the value can change explicitly with π , and it can also change because the boundary point moves through the ambient field. Keeping only the normal motion of the boundary gives

$$D_{\text{restricted}}(x, \pi) = \underbrace{\dot{g}_{\mathbb{R}^d}(x, \pi)}_{\text{explicit data change}} + \underbrace{\frac{\partial g}{\partial \vec{n}}(x, \pi) V_{\vec{n}}(x, \pi)}_{\text{motion through the field}} . \quad (6.5)$$

Mapped Boundary Data. Instead of defining the boundary data as a field in world space, we can also attach it directly to the surface, as one would with a texture map. In this case, the boundary data is defined on a fixed reference surface $\mathcal{M}$ and then transported to the moving boundary. Let $\Phi(\cdot, \pi) : \mathcal{M} \rightarrow \mathbb{R}^d$ parameterize the moving boundary, and let $\Psi(\cdot, \pi) : \partial\Omega(\pi) \rightarrow \mathcal{M}$ map a deformed boundary point back to its reference coordinate. A texture map or prescribed scalar field $g_{\mathcal{M}}$ assigns a boundary value to each reference point on $\mathcal{M}$, so the boundary value at $x \in \partial\Omega(\pi)$ is

$$g(x, \pi) = g_{\mathcal{M}}(\Psi(x, \pi), \pi). \quad (6.6)$$

Differentiating this nested expression is again just the chain rule: the texture value can change explicitly with π , and it can also change because the moving boundary point maps back to a different reference coordinate. Including the same normal motion term as above gives

$$D_{\text{mapped}}(x, \pi) = \underbrace{\dot{g}_{\mathcal{M}}(\Psi(x, \pi), \pi)}_{\text{explicit data change}} + \underbrace{\nabla g_{\mathcal{M}}(\Psi(x, \pi), \pi) \cdot \dot{\Psi}(x, \pi)}_{\text{motion through the texture}} + \underbrace{\frac{\partial g}{\partial \vec{n}}(x, \pi) V_{\vec{n}}(x, \pi)}_{\text{motion through the field}} . \quad (6.7)$$

Here the first two terms come from differentiating the nested texture lookup $g_{\mathcal{M}}(\Psi(x, \pi), \pi)$, while the last term is the same boundary-motion correction as in the restricted case.

6.2 First-Order Linear Boundary Conditions

The discussion above focuses on Dirichlet boundaries whose geometry depends on π . Other boundary conditions can be added with little extra work when they are imposed on fixed, non-parameterized parts of the boundary. Since these boundary conditions do not vary with π , differentiating them simply sets the corresponding boundary data for $\dot{u}$ to zero. For example, if a Robin boundary $\partial\Omega_R$ is independent of π and the primal solution satisfies

$$\frac{\partial u}{\partial \vec{n}}(x, \pi) - \mu u(x, \pi) = h(x) \quad \text{on } \partial\Omega_R, \quad (6.8)$$

with h and μ independent of π , then the derivative satisfies the homogeneous Robin condition

$$\frac{\partial \dot{u}}{\partial \vec{n}}(x, \pi) - \mu \dot{u}(x, \pi) = 0 \quad \text{on } \partial\Omega_R. \quad (6.9)$$

Thus fixed Robin boundaries can be handled by replacing walk on spheres with walk on stars, while the nested estimator remains unchanged: nesting enters only through the moving Dirichlet boundary data.

Parameterized Robin boundaries are more involved. Differentiating Robin data on a moving boundary is a standard shape-calculus calculation [Henrot and Pierre 2018], but it introduces higher-order spatial derivatives of the primal solution. We do not treat this case in this thesis. Recent work by Yu et al. [2025] develops estimators for this setting, using a generalized walk-on-stars procedure to evaluate the required higher derivatives and caching boundary-condition values to avoid naively branching a new walk at every step.

6.3 Differential Walk on Spheres

Starting from $x_0 = x$, each interior point x_k samples the next point x_{k+1} according to the Poisson-kernel transition used by WoS or WoSt. With this choice, the path throughput is one; for screened problems ($\sigma > 0$), the damped kernel $P^\sigma = Q^\sigma P$ of Equation (2.13) carries total mass $Q^\sigma < 1$, so each step additionally survives a Russian-roulette test with probability Q^σ , absorbing the screening attenuation into termination while keeping the throughput at one. The walk terminates when it reaches the ε -shell $\partial\Omega_\varepsilon$ of Section 2.3, where its closest boundary point is used to evaluate the boundary data. In the presence of Robin boundary conditions, Russian roulette additionally terminates the walk. The differential estimator mirrors the primal estimator [Muller 1956; Sawhney and Crane 2020] of Section 2.3, with the same recursion but new boundary data—and, since the differential problem is homogeneous, no source contribution:

Walk on spheres estimator

$$\hat{u}(x_k) := \begin{cases} g(\bar{x}_k), & x_k \in \partial\Omega_\varepsilon, \\ \hat{u}(x_{k+1}) + \frac{G^\sigma(x_k, z_k)}{p(z_k)} f(z_k), & \text{otherwise,} \end{cases}$$

Differential walk on spheres estimator

$$\hat{u}(x_k) := \begin{cases} D(\bar{x}_k, \pi) - V_{\vec{n}}(\bar{x}_k, \pi) \frac{\widehat{\partial u}}{\partial \vec{n}}(\bar{x}_k, \pi), & x_k \in \partial\Omega_\varepsilon, \\ \hat{u}(x_{k+1}), & \text{otherwise,} \end{cases} \quad (6.10)$$

where $\bar{x}_k$ denotes the closest point on $\partial\Omega$ to x_k , and $z_k \sim p$ is a source sample drawn within the ball around x_k . The boundary value includes the primal normal derivative $\partial u / \partial \vec{n}$ at the terminal point.

Estimating the Primal Normal Derivative. We estimate this quantity with a finite-difference stencil against the prescribed boundary value: from the terminal boundary point, we launch a primary walk at an inward offset $l > \varepsilon$ and form the backward difference (Figure 6.3, left)

$$\frac{\widehat{\partial u}}{\partial \vec{n}}(x, \pi) = \frac{g(x, \pi) - \hat{u}(x - l\vec{n}_x, \pi)}{l}. \quad (6.11)$$

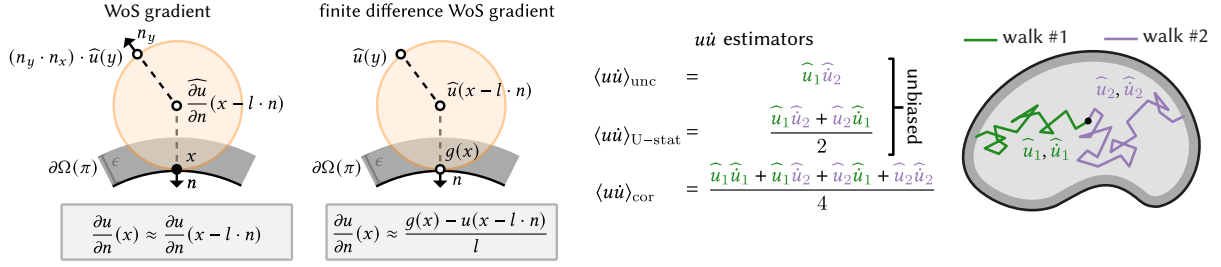


Figure 6.3: (Left) Estimating the primal normal derivative: the pointwise WoS gradient estimator evaluated at an offset point, versus the backward-difference stencil of Equation (6.11), which differences the prescribed boundary value against a primal estimate at the offset. (Right) Estimating the product $u\dot{u}$: reusing the same walk for both factors correlates the estimates and biases the product, while using each walk for only one factor is unbiased but wasteful; the U-statistic averages all cross-walk pairs, remaining unbiased while reusing every walk.

Here $\hat{u}$ is the primal Monte Carlo estimator of Chapter 2, realized with WoS or walk on stars (WoSt) depending on the boundary conditions, and the offset l trades bias against variance. Altogether, this gives a nested estimator: a differential walk first traces to the ε -shell, and then a primary walk launched from the offset point estimates the normal derivative needed to evaluate the differential boundary condition. We typically perform several primary walks per normal-derivative estimate, since these walks are inexpensive and normal-derivative estimates tend to have higher variance than solution estimates.

Unbiased Product Estimates. In practice, evaluating the shape-functional derivative of Equation (6.2) requires both u and $\dot{u}$. A single traced path can supply either quantity, since the primal and differential PDEs share the same domain and differ only in their source and boundary data. For a squared loss, however, the derivative contains a product of estimates, such as $u\dot{u}$. Reusing the same random walk for both factors gives a biased product estimator, since $\mathbb{E}[\hat{u}\hat{u}] \neq \mathbb{E}[\hat{u}]\mathbb{E}[\hat{u}]$ in general.

We remove this bias using a U-statistic estimator [Lee 1990] (Figure 6.3, right). Given N walks, let $\hat{u}_n$ and $\hat{\dot{u}}_n$ denote the primal and differential estimates obtained from the n th walk. The unbiased product estimate averages over all ordered pairs of distinct walks, excluding only the self-correlated pairs:

$$\langle u\dot{u} \rangle_{\text{U}} = \frac{1}{N(N-1)} \sum_{n=1}^N \hat{u}_n (S - \hat{u}_n), \quad S = \sum_{n=1}^N \hat{u}_n. \quad (6.12)$$

This expression is equivalent to the full pairwise average but can be computed in $O(N)$ time. The tradeoff is memory: the per-walk estimates $\hat{u}_n$ and $\hat{\dot{u}}_n$ must be stored until the statistic is formed. In implementation, we apply the U-statistic in batches of $B < N$ walks, accumulating Equation (6.12) after each batch. As the sample count grows, the bias of the naive correlated estimator becomes small, so in large-sample regimes one can also share walks directly and omit this correction.

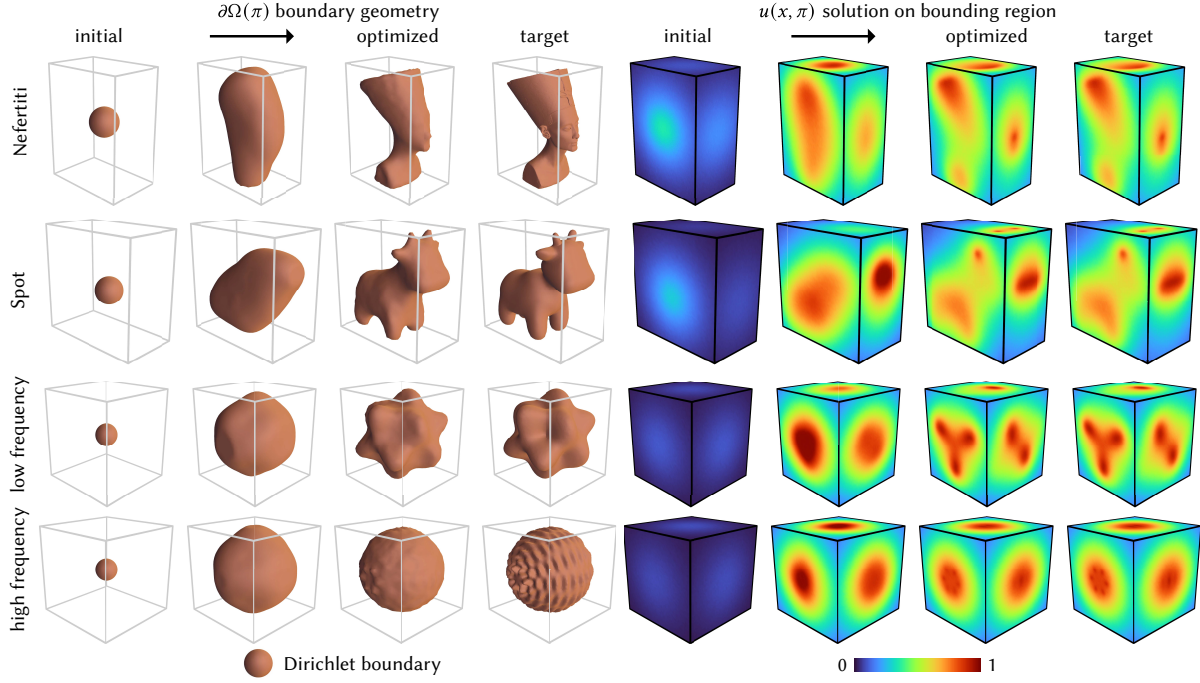


Figure 6.4: Shape from diffusion: starting from a sphere, boundary geometry evolves to match solution measurements on a surrounding box. Because screened diffusion low-passes the boundary signal, high-frequency surface detail leaves no measurable imprint and is not recovered (bottom row).

6.4 Evaluation

We demonstrate differential walk on spheres across several optimization settings, chosen to exercise its output sensitivity, tolerance of noisy gradients (Figure 6.6), and indifference to boundary representation.

Shape from Diffusion. We begin with an inverse reconstruction task, shape from diffusion, in which we recover the shape of a Dirichlet boundary in a screened Poisson problem using only measurements of the solution on a surrounding observation region. This setting mirrors diffuse optical tomography [Arridge 1999], where one infers the shape of an interior structure—a tumor, for example—from how it perturbs a diffusion profile measured at the surface. As shown in Figure 6.4, we obtain a plausible reconstruction while evaluating the solution only at measurement locations, never solving globally, and driving the optimization with noisy stochastic-gradient steps rather than exact gradients. The reconstruction is necessarily limited: screened diffusion acts as a low-pass filter on the boundary signal, so fine geometric details that leave no measurable imprint on the observed profile cannot be recovered, and a low-frequency approximation can reproduce the measurements just as well.

Thermal Design. We next consider optimization-driven thermal design, a setting where elliptic PDEs naturally arise from steady-state heat transfer. Following prior work on thermal system design [Zhan et al. 2008], we optimize the geometry of heating elements so that the

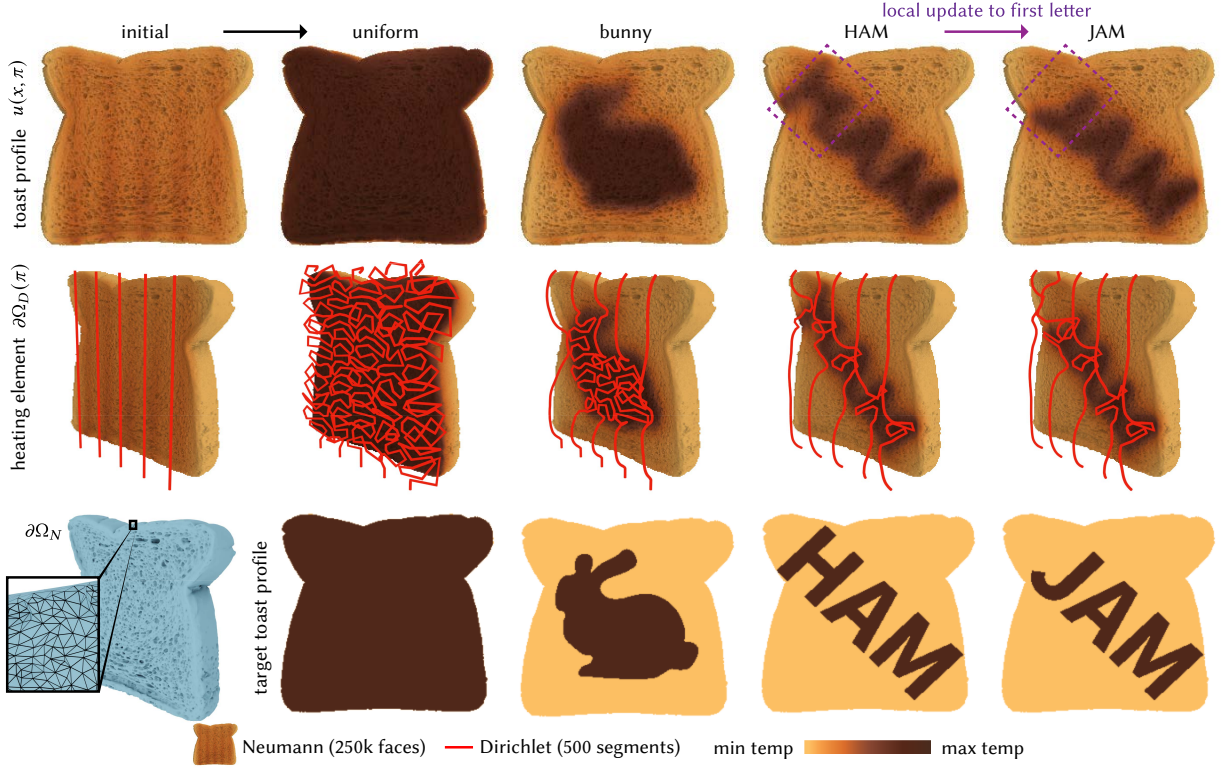


Figure 6.5: Optimization-driven thermal design: the heating elements of a toaster (Dirichlet boundary) are optimized so that the induced temperature field matches a target profile on a slice of bread (Neumann boundary). For partial updates to the target, derivatives are computed only over the modified region.

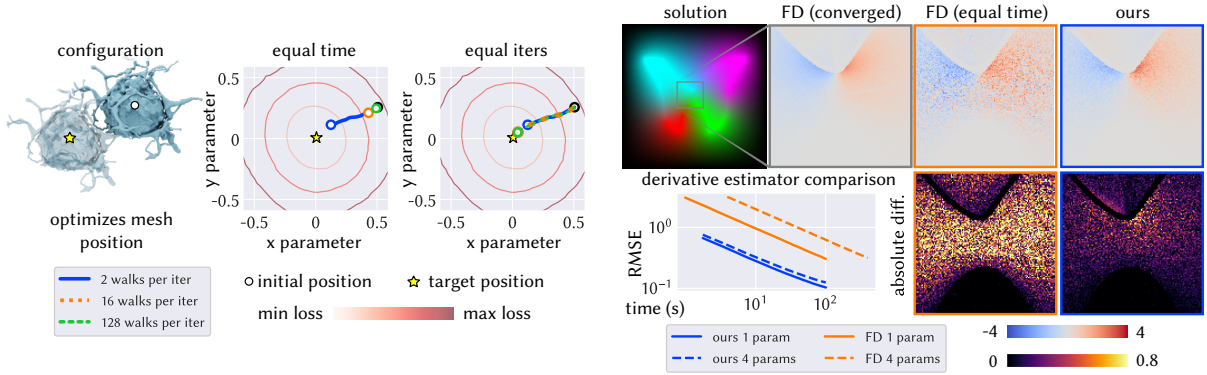


Figure 6.6: (Left) Noisy derivatives suit stochastic optimization: at equal time, cheap noisy estimates make more progress toward the target, while at equal iterations refined estimates land more precisely—inviting a schedule that increases the sample count as the iterate approaches a solution. (Right) Finite differences require a separate primal solve per parameter, whereas a single collection of nested walks estimates derivatives for all parameters simultaneously: at equal time the derivative noise is lower, and the gap widens with parameter count.

induced temperature field matches a target profile on an output surface (Figure 6.5). This example

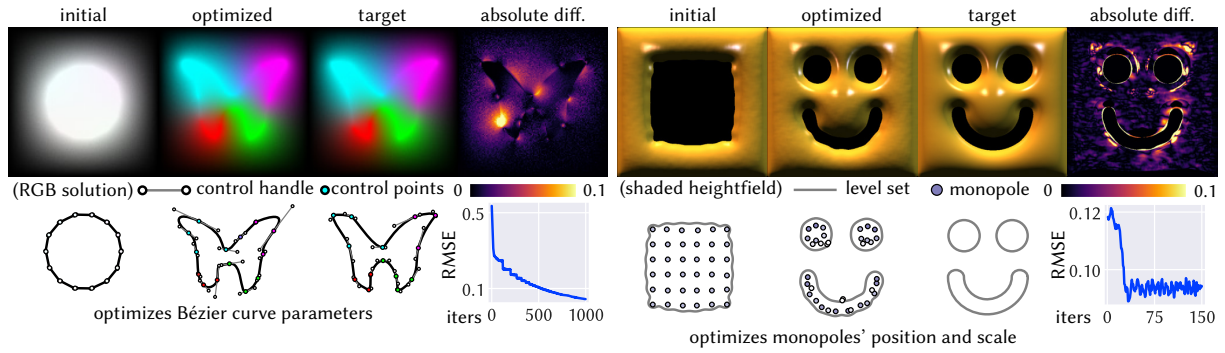


Figure 6.7: Optimizing geometric representations that conventional solvers cannot handle directly. (Left) Inverse diffusion curves: the control points, tangent handles, and RGB boundary data of Bézier curves are optimized jointly to match a target image. (Right) Shape inflation with an implicit boundary: the positions and scales of harmonic radial basis functions are optimized to match a target heightfield, with topological changes coming for free.

highlights output sensitivity: the loss is evaluated only where the target profile is observed, rather than over the entire simulation domain or all mesh elements. Differential walk on spheres therefore spends computation only on the degrees of freedom that affect the objective, making it well suited to local edits and design problems where the quantity of interest is much smaller than the full state.

Alternative Geometric Representations. Finally, we optimize boundary representations that conventional solvers cannot handle directly, parameterizing the geometry itself and avoiding any fixed volumetric discretization (Figure 6.7). The first is an inverse diffusion-curves problem [Zhao et al. 2017], where we jointly optimize the control points, tangent handles, and RGB boundary data of Bézier curves to match a target image, using a Monte Carlo length regularizer to discourage overly complex curves. The second is a shape-inflation problem [Baran and Lehtinen 2009] with an implicit boundary written as a sum of harmonic radial basis functions, where we optimize monopole positions and scales to match a target heightfield. Large topological changes come for free, closest-point queries follow from Harnack bounds [Gillespie et al. 2024], and the boundary integral in Equation (6.2) is approximated by a thin-shell volume integral since the implicit boundary cannot be sampled directly.

6.5 Outlook

Reusing Primal-Solver and Rendering Machinery. The foundation laid here is deliberately simple. By casting differentiation as a nested boundary value problem, the differential solution can be estimated with a small modification to the primal recursive solver: when the differential walk reaches the boundary, it calls a primal estimator to evaluate the boundary data. Because this primal estimator is treated as a black box, much of the machinery built around forward Monte Carlo solvers carries over with little change. This includes variance-reduction and caching schemes from the WoS literature [Qi et al. 2022; Miller et al. 2023; Bakbouk and Peers 2023], as

well as optimization-loop techniques from differentiable rendering, such as recursive control variates that reuse gradient information across descent steps [Nicolet et al. 2023]. Adapting these techniques to the differential boundary value problem setting remains an open direction for future work.

Alternative Approaches to Differential Estimators. Concurrent work arrives at related estimators for Dirichlet problems through different derivations [Yu et al. 2024; Yilmazer et al. 2024]. The advantage of the nested boundary value problem perspective is that it makes the structural connection to the primal solver explicit: the derivative solves another boundary value problem on the same domain, using the same geometric queries, with new boundary data coupled to the primal solution. As a result, extensions that might otherwise look like new estimator derivations become adaptations of existing forward-solver machinery. Fixed Robin boundaries, for example, are handled by replacing WoS with WoSt, and subsequent work has used the same template to differentiate with respect to parameters of Neumann boundaries [Yu et al. 2025]. The same reuse underlies the range of results in this chapter. Because differentiation changes only the boundary data, the differential solver inherits the primal solver’s geometric scalability and representation-agnosticism. The three-dimensional shape-optimization examples across meshes, splines, and implicit surfaces thus come from a single forward solver rather than more specialized estimators.

Finding the Defining Application. Finally, differentiable rendering found its defining application in material and surface reconstruction from images. Differential Monte Carlo PDE solvers arguably face the opposite—and more promising—situation: shape optimization is ubiquitous across engineering and the sciences, so candidate applications abound. The open work lies in identifying where the distinctive features of a Monte Carlo solver are most valuable. For instance, inverse problems with sparse measurement probes or design objectives localized to small regions of the domain can benefit from the solver’s output sensitivity, which spends computation only where the objective is supported. Optimization over native geometric representations—implicit functions, splines, or neural fields—can likewise benefit from its indifference to representation, avoiding any conversion to a separate grid or simulation mesh. Matching these advantages to the problems where they matter most is an exciting next step for differential Monte Carlo methods.

Chapter 7

Future Directions

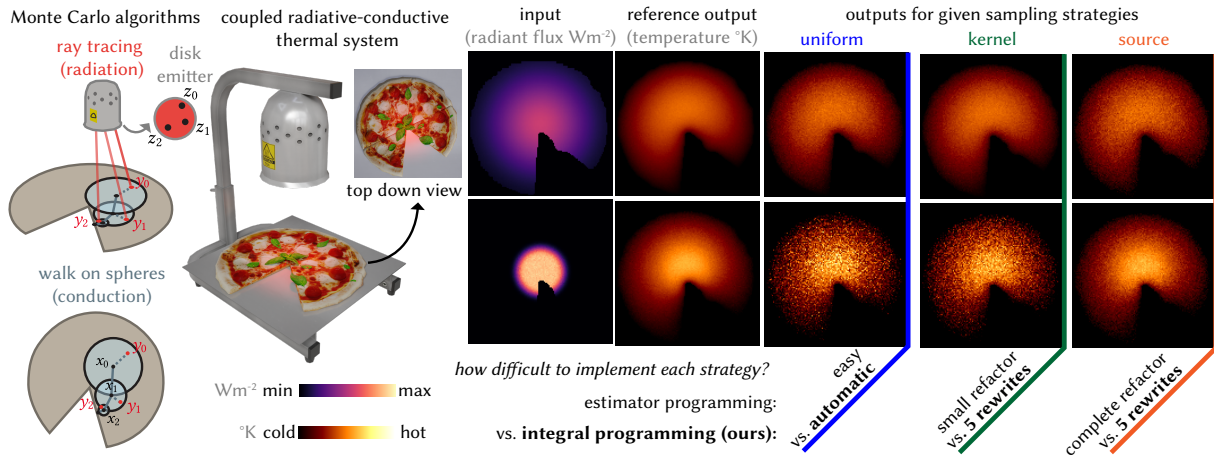


Figure 7.1: A heat lamp radiatively warms a pizza while conduction redistributes heat across its surface, modeled as a coupled system of integral equations—radiative transfer evaluated by ray tracing, conduction by walk on spheres (*left*). Effective sampling strategies for the coupled system interleave operations from both algorithms, which ordinarily requires merging two solvers into one monolithic estimator program; expressed as rewrites of the composed integral program, the same strategies take a few lines, and remain robust as heating profiles concentrate (*right*). Figure reproduced from Mu et al. [2026], work currently under submission.

This thesis develops new capabilities for walk on spheres, taking inspiration from those central to Monte Carlo rendering. The Introduction asked why such developments make sense to pursue now; the chapters that followed pursued them. It seems appropriate to close by asking what comes after—not by returning to rendering’s earliest history, but by looking at the directions rendering has itself evolved into over the past six years, and at the foundational tooling we have not yet adapted. Three frontiers stand out, each picking up a through-line of the dissertation.

7.1 DSLs for Recursive Integral Equations

Every chapter began with a classical integral equation, performed a small number of principled rewrites, and read off a Monte Carlo estimator. That workflow was carried out by hand throughout the dissertation. Rendering, by contrast, has long benefited from programming languages and domain-specific languages that codify such workflows at scale—from Halide’s separation of algorithm and schedule for image-processing pipelines [Ragan-Kelley et al. 2013], to Aether’s embedded language for Monte Carlo rendering [Anderson et al. 2017]. A natural next step for Monte Carlo PDE solving is the analogous substrate: a language in which integral equations are programs and estimator strategies are equivalence-preserving rewrites, yielding correct-by-construction estimators and a programmable record of the derivations that justify them. Martingale [Mu et al. 2026], work of ours currently under submission, is an early step in this direction. Figure 7.1 shows an example: a coupled thermal problem whose estimator must interleave ray tracing and walk on spheres. By hand, that means merging two solvers into one program; as rewrites of the composed integral equations, it takes a few lines and stays correct by construction. Because the rewrites act on the composed program, they can freely intermix the two solvers—crossing a compositional boundary that a rendering engine paired with a PDE solver does not easily allow.

7.2 Stochastic Geometry $\times$ Monte Carlo PDE Solvers

The participating-media chapter adapted volume rendering’s stochastic geometry for microparticle media. Rendering has since extended these models to stochastic solid geometry [Miller et al. 2024; Seyb et al. 2024], where the uncertainty lies in the surface or solid itself rather than in the positions of microparticles (Figure 7.2). Here the stochastic geometry $\mathcal{O}$ is a solid described by a random indicator rather than a union of sampled particles,

$$\mathcal{O} := \{x \in \mathbb{R}^3 \mid I(x) = 1\}, \quad o(x) := \Pr[I(x) = 1], \quad v := 1 - o, \quad (7.1)$$

and under a Markov assumption along rays, the distance to the first intersection follows the same exponential free-flight law as the Poisson Boolean model of Section 2.5,

$$p_{x,\vec{\omega}}^{\text{ff}}(t) = \sigma(r_{x,\vec{\omega}}(t), \vec{\omega}) \exp\left(-\int_0^t \sigma(r_{x,\vec{\omega}}(s), \vec{\omega}) \, ds\right), \quad \sigma(x, \vec{\omega}) = \frac{|\vec{\omega} \cdot \nabla v(x)|}{v(x)}, \quad (7.2)$$

with the rate now derived from surface uncertainty rather than particle density [Miller et al. 2024]. The same lens may extend to primitive-based representations. Splatting methods represent a scene not as a field but as a large collection of simple geometric primitives—for example, the disk-shaped surfels of 2D Gaussian splatting [Huang et al. 2024]—optimized directly from images. Each primitive carries a small set of parameters, and a natural starting point is to let some of those parameters be uncertain. A collection of disks whose centers are Gaussian-distributed,

$$\mathcal{O} := \bigcup_{i=1}^N D(c_i, \vec{n}_i, R_i), \quad c_i \sim \mathcal{N}(\tilde{c}_i, \Sigma_i), \quad (7.3)$$

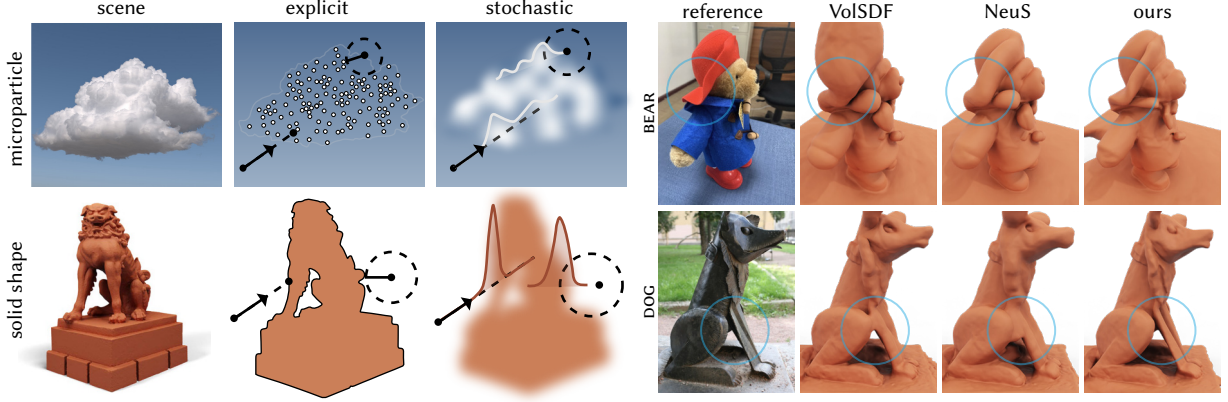


Figure 7.2: (Left) Stochastic opaque solids extend solid geometry with uncertainty, much as microparticle models capture geometric complexity, replacing deterministic queries with distributional equivalents. (Right) Probabilistic free-flight queries on uncertain solids improve neural surface reconstruction over more heuristic volumetric models [Miller et al. 2024]; a closest-point analogue of the same query would unlock walk on spheres on these representations.

where $D(c, \vec{n}, R)$ denotes a disk with center c , normal $\vec{n}$, and radius R , is a stochastic geometry in its own right, and one could equally place distributions on normals or other parameters. The questions then mirror the field model above: How does one compute transmittance and free-flight distributions under such a model, and does the resulting uncertainty benefit surface reconstruction as it does in the implicit setting [Miller et al. 2024]? Answers here would carry the volumetric theory underlying the field models into splatting-style representations, which have become enormously popular in reconstruction. More broadly, an interesting future direction is to adapt volumetric walk on spheres to such stochastic geometries—the missing ingredient is the closest-point analogue of the free-flight query—which would enable analysis of uncertain solids that naturally arise in cases such as engineering tolerances or processing scanned geometry [Genest and Coeurjolly 2026; Sellán and Jacobson 2022].

7.3 Neural Fields $\times$ Monte Carlo PDE Solvers

Recent rendering work has turned heavily toward neural-field methods such as NeRF [Mildenhall et al. 2020] and 3D Gaussian splatting [Kerbl et al. 2023] for reconstruction, with the most successful of these methods built on volume rendering. PDEs and neural fields already meet in three ways (Figure 7.3)—solving PDEs on neural fields, representing neural fields with PDEs, and processing neural fields with PDEs,

$$\underbrace{\begin{aligned} \Delta u &= 0 && \text{in } \Omega_{\text{rec}}, \\ u &= g && \text{on } \partial\Omega_{\text{rec}}, \end{aligned}}_{\text{solving on fields}} \quad \underbrace{v(x) = \Phi\left(s\left(\frac{1}{2} - \sum_{i=1}^N a_i P_\epsilon(x, p_i, \vec{n}_i)\right)\right)}_{\text{representing fields}} \quad \underbrace{g_S = (\text{Id} - \tau\Delta)^{-1}g}_{\text{processing fields}}. \quad (7.4)$$

The first poses a Laplace problem on a domain bounded by reconstructed Gaussian surfels, recovering the interior temperature of an object from multi-view thermal images without ever

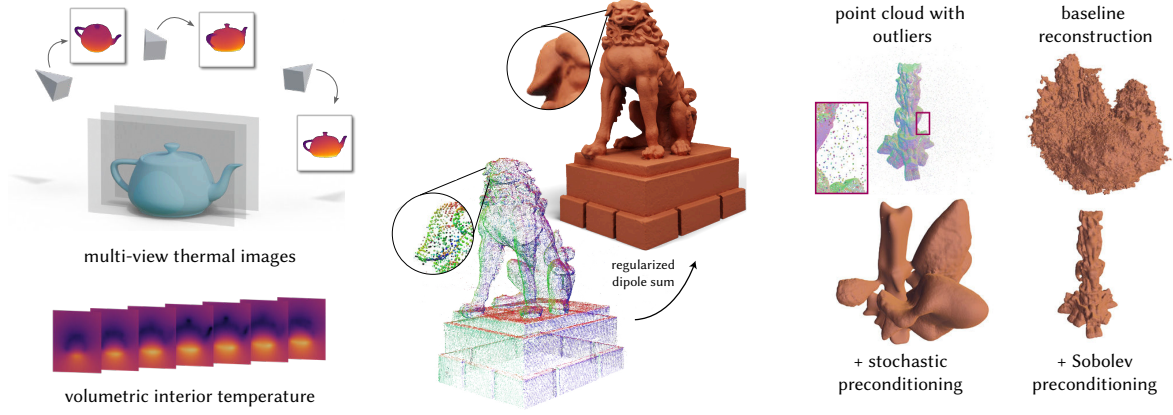


Figure 7.3: The three ways PDEs and neural fields already meet, illustrated left to right. *(Left)* Solving a PDE on a neural field, conductive thermal Gaussian surfels reconstruct both the surface and interior temperature of an object from multi-view thermal images [Liu et al. 2026]. *(Center)* Representing a neural field with PDE machinery, a regularized dipole sum turns a structure-from-motion point cloud into a point-based implicit surface and radiance field optimized by differentiable rendering [Chen et al. 2024a]. *(Right)* Processing a neural field through a PDE solve, Sobolev preconditioning recovers a clean surface from an outlier-ridden point cloud, where the unpreconditioned baseline and an alternative stochastic preconditioning either fail or oversmooth [Su et al. 2026].

meshing it, in work of ours currently under submission [Liu et al. 2026]. The second generalizes the winding number—the harmonic machinery behind Poisson surface reconstruction [Kazhdan et al. 2006]—into a point-based geometry and radiance representation with fast forward and adjoint queries [Chen et al. 2024a]. The third smooths the noisy gradients of neural implicit geometry through a screened-Poisson solve during backpropagation, again in work of ours under submission [Su et al. 2026]. Each direction also has a differentiable form—optimizing through the solve recovers boundaries [Yilmazer et al. 2024; Narayanan et al. 2024], through the representation fits the field, and through the processing operator learns the diffusion itself. Across these three directions, the grid-free Monte Carlo solvers of this thesis have yet to be fully incorporated. More broadly, PDE-based machinery recurs throughout geometry processing and learning, from DiffusionNet’s learned diffusion layers [Sharp et al. 2022] to recent Monte Carlo estimation of Steklov operators built directly on walk on spheres [Maesumi et al. 2026]. Much of neural rendering’s success along these lines has moreover come from shifting to volumetric methods, which essentially incorporate the stochastic geometric representations of the previous section; making the same shift here is an appealing direction for future work.

7.4 Closing Remarks

Monte Carlo geometry processing [Sawhney and Crane 2020] first drew the analogy between rendering and walk on spheres; the goal of this thesis has been to demonstrate how far that analogy extends. By adapting rendering’s treatment of materials, participating media, sample reuse, and differentiability, we developed support for general first-order linear boundary con-

ditions (Chapter 3), generalizations of walk on spheres to participating media (Chapter 4), a caching scheme that enables global reuse across evaluation points (Chapter 5), and differentiable variants for inverse and design problems (Chapter 6). Together, these developments carry walk on spheres beyond a curious novelty and toward a maturing, though still nascent, solver framework, one whose adoption across computer graphics continues to grow [Sawhney et al. 2025]. The frontiers above suggest that rendering’s more recent past holds further capabilities worth adapting, and because the two methods share the same underlying mathematical structure, much of that machinery can be carried over directly rather than reinvented.

More broadly, the conditions that motivate Monte Carlo solvers continue to strengthen. The geometry arising in scientific and visual computing continues to grow in complexity and noise, increasingly parallel hardware continues to favor trivially parallelizable methods, and developments in adjacent fields—such as the learned denoising and caching now standard in rendering—stand to make these methods more capable still. With continued development, we believe grid-free Monte Carlo methods will mature into a solver framework that makes simulation, design, and inverse problems as routine on complex real-world geometry as rendering has made image synthesis on complex scenes.

Bibliography

- David Andelman. Electrostatic properties of membranes: the Poisson-boltzmann theory. In *Handbook of biological physics*, volume 1, pages 603–642. Elsevier, 1995.
- Luke Anderson, Tzu-Mao Li, Jaakko Lehtinen, and Frédo Durand. Aether: An embedded domain specific sampling language for Monte Carlo rendering. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 36(4):99:1–99:16, 2017. doi: 10.1145/3072959.3073704.
- Simon R Arridge. Optical tomography in medical imaging. *Inverse problems*, 1999.
- James Richard Arvo. *Analytic methods for simulated light transport*. PhD thesis, Yale University, 1995.
- Ghada Bakbouk and Pieter Peers. Mean value caching for walk on spheres. In *Eurographics Symposium on Rendering*, June 2023. doi: <https://doi.org/10.2312/sr.20231120>.
- Anchang Bao, Enya Shen, and Jianmin Wang. Monte Carlo PDE solvers for nonlinear radiative boundary conditions. *arXiv preprint arXiv:2604.21717*, 2026.
- Ilya Baran and Jaakko Lehtinen. Notes on inflating curves. Technical report, MIT, 2009.
- Gavin Barill, Neil G Dickson, Ryan Schmidt, David IW Levin, and Alec Jacobson. Fast winding numbers for soups and clouds. *ACM Transactions on Graphics*, 37(4):1–12, 2018.
- Josh Barnes and Piet Hut. A hierarchical $O(n \log n)$ force-calculation algorithm. *nature*, 324(6096):446–449, 1986.
- Mégane Bati, Stéphane Blanco, Christophe Coustet, Vincent Eymet, Vincent Forest, Richard Fournier, Jacques Gautrais, Nicolas Mellado, Mathias Paulin, and Benjamin Piaud. Coupling conduction, convection and radiative transfer in a single path-space: Application to infrared rendering. *ACM Transactions on Graphics*, 42(4):1–20, 2023.
- Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of machine learning research*, 18(153):1–43, 2018.
- Rick Beatson, Leslie Greengard, et al. A short course on fast multipole methods. *Wavelets, multilevel methods and elliptic PDEs*, 1(1), 1997.
- Yash Belhe, Bing Xu, Sai Praveen Bangaru, Ravi Ramamoorthi, and Tzu-Mao Li. Importance sampling BRDF derivatives. *ACM Transactions on Graphics*, 2024.
- Angelica Bianco, Monica Passananti, Marcello Brigante, and Gilles Mailhot. Photochemistry of the cloud aqueous phase: A review. *Molecules*, 25(2):423, 2020.

- Ilia Binder and Mark Braverman. The rate of convergence of the walk on spheres algorithm. *Geometric and Functional Analysis*, 22(3):558–587, 2012.
- Benedikt Bitterli, Srinath Ravichandran, Thomas Müller, Magnus Wrenninge, Jan Novák, Steve Marschner, and Wojciech Jarosz. A radiative transfer framework for non-exponential media. *ACM Trans. Graph.*, 37(6), December 2018. ISSN 0730-0301. doi: 10.1145/3272127.3275103. URL <https://doi.org/10.1145/3272127.3275103>.
- Benedikt Bitterli, Chris Wyman, Matt Pharr, Peter Shirley, Aaron Lefohn, and Wojciech Jarosz. Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 39(4), July 2020. doi: 10/gg8xc7.
- S. Chandrasekhar. *Radiative Transfer*. Dover Books on Physics. Dover Publications, 2013. ISBN 9780486318455. URL <https://books.google.com/books?id=1YHCagAAQBAJ>.
- Hanyu Chen, Bailey Miller, and Ioannis Gkioulekas. 3D reconstruction with fast dipole sums. *ACM Trans. Graph.*, 43(6), November 2024a. ISSN 0730-0301. doi: 10.1145/3687914. URL <https://doi.org/10.1145/3687914>.
- Jiong Chen, Florian Schäfer, and Mathieu Desbrun. Lightning-fast method of fundamental solutions. *ACM Transactions on Graphics*, 43(4), 2024b.
- Jiong Chen, Florian Schäfer, and Mathieu Desbrun. Lightning-fast boundary element method. *ACM Transactions on Graphics*, 44(4), 2025.
- Sung Nok Chiu, Dietrich Stoyan, Wilfrid S Kendall, and Joseph Mecke. *Stochastic geometry and its applications*. John Wiley & Sons, 2013.
- Martin Costabel. Principles of boundary element methods. *Computer Physics Reports*, 6(1-6): 243–274, 1987.
- Roger R Coveyou, VR Cain, and KJ Yost. Adjoint and importance in Monte Carlo application. *Nuclear Science and Engineering*, 27(2):219–234, 1967.
- Michael Czekanski, Benjamin Faber, Margaret Fairborn, Adelle Wright, and David Bindel. Walking on spheres and talking to neighbors: Variance reduction for Laplace’s equation. *arXiv preprint arXiv:2404.17692*, 2024.
- Carsten Dachsbacher, Jaroslav Křivánek, Miloš Hašan, Adam Arbree, Bruce Walter, and Jan Novák. Scalable realistic rendering with many-light methods. In *Computer Graphics Forum*, volume 33, pages 88–104. Wiley Online Library, 2014.
- Fernando de Goes and Mathieu Desbrun. Stochastic computation of barycentric coordinates. *ACM Transactions on Graphics*, 43(4), 2024. doi: 10.1145/3658131.
- Robert A. Drebin, Loren Carpenter, and Pat Hanrahan. Volume rendering. *SIGGRAPH Comput. Graph.*, 22(4):65–74, June 1988. ISSN 0097-8930. doi: 10.1145/378456.378484. URL <https://doi.org/10.1145/378456.378484>.
- Dean G. Duffy. *Green’s Functions with Applications*. Chapman and Hall/CRC, 2nd edition, 2015.
- Eugene d’Eon. A reciprocal formulation of nonexponential radiative transfer. 1: Sketch and motivation. *Journal of Computational and Theoretical Transport*, 47(1-3):84–115, 2018. doi: 10.1080/23324309.2018.1481433. URL <https://doi.org/10.1080/23324309.2018.1481433>.

- Sergey M Ermakov and Alexander S Sipin. The “walk in hemispheres” process and its applications to solving boundary value problems. *Vestnik St. Petersburg University: Mathematics*, 42:155–163, 2009. ISSN 1934-7855.
- Lawrence C. Evans. *Partial Differential Equations*, volume 19 of *Graduate Studies in Mathematics*. American Mathematical Society, 1998.
- Bruce C Faust. Photochemistry of clouds, fogs, and aerosols. *Environmental science & technology*, 28(5):216A–222A, 1994.
- Baptiste Genest and David Coeurjolly. Uncertainty-aware geometry processing on Gaussian process implicit surfaces. *ACM Transactions on Graphics*, 45(4), 2026.
- Iliyan Georgiev, Zackary Misso, Toshiya Hachisuka, Derek Nowrouzezahrai, Jaroslav Krivánek, and Wojciech Jarosz. Integral formulations of volumetric transmittance. *ACM Trans. Graph.*, 38(6), November 2019. ISSN 0730-0301. doi: 10.1145/3355089.3356559. URL <https://doi.org/10.1145/3355089.3356559>.
- Mark Gillespie, Denise Yang, Mario Botsch, and Keenan Crane. Ray tracing harmonic functions. *ACM Trans. Graph.*, 43(4), July 2024. ISSN 0730-0301. doi: 10.1145/3658201. URL <https://doi.org/10.1145/3658201>.
- Arianna Giunti, Richard Höfer, and Juan J. L. Velázquez. Homogenization for the Poisson equation in randomly perforated domains under minimal assumptions on the size of the holes. *Communications in Partial Differential Equations*, 43(9):1377–1412, 2018. doi: 10.1080/03605302.2018.1531425. URL <https://doi.org/10.1080/03605302.2018.1531425>.
- Ioannis Gkioulekas, Shuang Zhao, Kavita Bala, Todd Zickler, and Anat Levin. Inverse volume rendering with material dictionaries. *ACM Trans. Graph.*, 32(6), November 2013. ISSN 0730-0301. doi: 10.1145/2508363.2508377. URL <https://doi.org/10.1145/2508363.2508377>.
- Ioannis Gkioulekas, Anat Levin, and Todd Zickler. An evaluation of computational imaging techniques for heterogeneous inverse scattering. In *European Conference on Computer Vision*, pages 685–701. Springer, 2016.
- Andreas Griewank and Andrea Walther. *Evaluating derivatives: principles and techniques of algorithmic differentiation*. SIAM, 2008.
- Toshiya Hachisuka and Henrik Wann Jensen. Stochastic progressive photon mapping. In *ACM SIGGRAPH Asia 2009 papers*, pages 1–8. 2009.
- Stefan Heinrich and Peter Mathé. The Monte Carlo complexity of Fredholm integral equations. *Mathematics of Computation*, 60(201):257–278, 1993.
- Eric Heitz, Jonathan Dupuy, Cyril Crassin, and Carsten Dachsbacher. The SGGX microflake distribution. *ACM Trans. Graph.*, 34(4), July 2015. ISSN 0730-0301. doi: 10.1145/2766988. URL <https://doi.org/10.1145/2766988>.
- Eric Heitz, Johannes Hanika, Eugene d’Eon, and Carsten Dachsbacher. Multiple-scattering microfacet BSDFs with the Smith model. *ACM Transactions on Graphics*, 35(4):1–14, 2016.
- Antoine Henrot and Michel Pierre. *Shape variation and optimization*, volume 28. European Mathematical Society Publishing House, Zuerich, Switzerland, 2018.

- Valerie N. P. Ho and Art B. Owen. Walk on spheres and array-rqmc, 2026a. URL <https://arxiv.org/abs/2605.12844>.
- Valerie N. P. Ho and Art B. Owen. Randomized quasi-Monte Carlo for walk on spheres, 2026b. URL <https://arxiv.org/abs/2605.08483>.
- Ethan Hsu, Hong Meng Yam, and Ivan Ge. Mc^2 : Monte Carlo correction for fast elliptic PDE solving, 2026. URL <https://arxiv.org/abs/2605.09288>.
- Yixin Hu, Teseo Schneider, Bolun Wang, Denis Zorin, and Daniele Panozzo. Fast tetrahedral meshing in the wild. *ACM Transactions on Graphics*, 39(4):117:1–117:18, 2020.
- Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2D Gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, 2024.
- Tianyu Huang. Geometric queries on closed implicit surfaces for walk on stars. In *Proceedings of the SIGGRAPH Asia 2025 Technical Communications*, SA Technical Communications ’25, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400721366. doi: 10.1145/3757376.3771378. URL <https://doi.org/10.1145/3757376.3771378>.
- Tianyu Huang, Jingwang Ling, Shuang Zhao, and Feng Xu. Guiding-based importance sampling for walk on stars. *Proceedings of the ACM SIGGRAPH Conference Papers (SIGGRAPH Conference Papers ’25)*, 2025. doi: 10.1145/3721238.3730593.
- Wanchao Huang, Yutian Zhu, Qing Fang, and Ligang Liu. Probe-based walk on spheres for efficient path reusing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 2026.
- Peter Hunter and Andrew Pullan. FEM/BEM notes, 2001. Department of Engineering Science, The University of Auckland, New Zealand.
- Alec Jacobson, Ladislav Kavan, and Olga Sorkine-Hornung. Robust inside-outside segmentation using generalized winding numbers. *ACM Trans. Graph.*, 32(4), July 2013. ISSN 0730-0301. doi: 10.1145/2461912.2461916. URL <https://doi.org/10.1145/2461912.2461916>.
- Pranav Jain, Ziyin Qu, Peter Yichen Chen, and Oded Stein. Neural Monte Carlo fluid simulation. In *ACM SIGGRAPH 2024 Conference Papers*, SIGGRAPH ’24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400705250. doi: 10.1145/3641519.3657438. URL <https://doi.org/10.1145/3641519.3657438>.
- Wenzel Jakob, Adam Arbree, Jonathan T. Moon, Kavita Bala, and Steve Marschner. A radiative transfer framework for rendering materials with anisotropic structure. *ACM Trans. Graph.*, 29(4), July 2010. ISSN 0730-0301. doi: 10.1145/1778765.1778790. URL <https://doi.org/10.1145/1778765.1778790>.
- Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, Merlin Nimier-David, Delio Vicini, Tizian Zeltner, Baptiste Nicolet, Miguel Crespo, Vincent Leroy, and Ziyi Zhang. Mitsuba 3 renderer, 2022. <https://mitsuba-renderer.org>.
- Clément Jambon, Mohammad Sina Nabizadeh, and Mina Konaković Luković. Walk on decomposed subdomains: A hybrid Monte Carlo–deterministic solver for elliptic PDEs. *ACM Trans. Graph.*, 45(4), July 2026. doi: 10.1145/3811340. URL <https://doi.org/10.1145/3811340>.

- Adrian Jarabo, Carlos Aliaga, and Diego Gutierrez. A radiative transfer framework for spatially-correlated materials. *ACM Trans. Graph.*, 37(4), July 2018. ISSN 0730-0301. doi: 10.1145/3197517.3201282. URL <https://doi.org/10.1145/3197517.3201282>.
- Henrik Wann Jensen. Global illumination using photon maps. In *Eurographics workshop on Rendering techniques*, pages 21–30. Springer, 1996.
- David E. Johnson and Elaine Cohen. Spatialized normal come hierarchies. In *Proceedings of the 2001 Symposium on Interactive 3D Graphics, I3D '01*, page 129–134, New York, NY, USA, 2001. Association for Computing Machinery. ISBN 1581132921. doi: 10.1145/364338.364380. URL <https://doi.org/10.1145/364338.364380>.
- Konrad Jörgens. *Linear integral operators*. Pitman, 1982.
- Herman Kahn. Random sampling (Monte Carlo) techniques in neutron attenuation problems–i. *Nucleonics*, 6(5):27–33, 1950.
- James T. Kajiya. The rendering equation. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '86*, page 143–150, New York, NY, USA, 1986. Association for Computing Machinery. ISBN 0897911962. doi: 10.1145/15922.15902. URL <https://doi.org/10.1145/15922.15902>.
- Malvin H. Kalos and Paula A. Whitlock. *Monte Carlo Methods*. Wiley-VCH, Weinheim, 2nd edition, 2008.
- Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing, SGP '06*, page 61–70, Goslar, DEU, 2006. Eurographics Association. ISBN 3905673363.
- Alexander Keller. Quasi-monte carlo radiosity. In *Eurographics Workshop on Rendering Techniques*, pages 101–110. Springer, 1996.
- Alexander Keller. Instant radiosity. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 49–56, 1997.
- Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023.
- Markus Kettunen, Eugene D'Eon, Jacopo Pantaleoni, and Jan Novák. An unbiased ray-marching transmittance estimator. *ACM Trans. Graph.*, 40(4), July 2021. ISSN 0730-0301. doi: 10.1145/3450626.3459937. URL <https://doi.org/10.1145/3450626.3459937>.
- Thomas Kollig and Alexander Keller. Illumination in the presence of weak singularities. In *Monte Carlo and Quasi-Monte Carlo Methods 2004*, pages 245–257. Springer, 2006.
- Peter Kutz, Ralf Habel, Yining Karl Li, and Jan Novák. Spectral and decomposition tracking for rendering heterogeneous volumes. *ACM Trans. Graph.*, 36(4), July 2017. ISSN 0730-0301. doi: 10.1145/3072959.3073665. URL <https://doi.org/10.1145/3072959.3073665>.
- Günter Last and Mathew Penrose. *Lectures on the Poisson process*, volume 7. Cambridge University Press, 2018.
- Alan J. Lee. *U-Statistics: Theory and Practice*. Routledge, 1st edition, 1990. doi: 10.1201/9780203734520. URL <https://doi.org/10.1201/9780203734520>.

- Peter A. W. Lewis and Gerald S. Shedler. Simulation of nonhomogeneous Poisson processes by thinning. *Naval Research Logistics Quarterly*, 26(3):403–413, 1979.
- Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable Monte Carlo ray tracing through edge sampling. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, 37(6):222:1–222:11, 2018.
- Zilu Li, Guandao Yang, Xi Deng, Christopher De Sa, Bharath Hariharan, and Steve Marschner. Neural caches for monte carlo partial differential equation solvers. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–10, 2023.
- Zhenyuan Liu, Bailey Miller, Bharath Seshadri, George Kopanas, Ioannis Gkioulekas, and Bernd Bickel. Conductive thermal Gaussian surfels. Under submission, 2026.
- Guillaume Loubet, Nicolas Holzschuch, and Wenzel Jakob. Reparameterizing discontinuous integrands for differentiable rendering. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)*, 38(6), December 2019. doi: 10.1145/3355089.3356510.
- Arman Maesumi, Tanish Makadia, Aruna Anderson, Oras Phongpanangam, Justin Solomon, and Daniel Ritchie. Monte Carlo Steklov operators for large-scale geometry processing in the wild, 2026.
- Sylvain Maire and Etienne Tanré. Monte Carlo approximations of the Neumann problem. *arXiv preprint arXiv:1203.4910*, 2012.
- Vladimir A Marchenko and Evgueni Ya Khruslov. *Homogenization of partial differential equations*, volume 46. Springer Science & Business Media, 2008.
- Michael Mascagni and Nikolai A Simonov. Monte Carlo methods for calculating some physical properties of large molecules. *SIAM Journal on Scientific Computing*, 26(1):339–357, 2004.
- Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.
- Bailey Miller, Iliyan Georgiev, and Wojciech Jarosz. A null-scattering path integral formulation of light transport. *ACM Trans. Graph.*, 38(4), July 2019. ISSN 0730-0301. doi: 10.1145/3306346.3323025. URL <https://doi.org/10.1145/3306346.3323025>.
- Bailey Miller, Rohan Sawhney, Keenan Crane, and Ioannis Gkioulekas. Boundary value caching for walk on spheres. *ACM Transactions on Graphics*, 42(4):1–11, July 2023. ISSN 1557-7368. doi: 10.1145/3592400. URL <http://dx.doi.org/10.1145/3592400>.
- Bailey Miller, Hanyu Chen, Alice Lai, and Ioannis Gkioulekas. Objects as volumes: A stochastic geometry view of opaque solids. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 87–97, June 2024.
- Kevin Mu, Bailey Miller, Ioannis Gkioulekas, and Zachary Tatlock. Martingale: Programming with integrals. Under submission, 2026.
- Mervin E Muller. Some continuous Monte Carlo methods for the Dirichlet problem. *The Annals of Mathematical Statistics*, pages 569–589, 1956.

- Sriram Narayanan, Mani Ramanagopal, Mark Sheinin, Aswin C Sankaranarayanan, and Srinivasa G Narasimhan. Shape from heat conduction. In *European Conference on Computer Vision*, pages 426–444. Springer, 2024.
- Jean-Claude Nédélec. *Acoustic and electromagnetic equations: integral representations for harmonic problems*, volume 144. Springer, 2001.
- Baptiste Nicolet, Fabrice Rousselle, Jan Novák, Alexander Keller, Wenzel Jakob, and Thomas Müller. Recursive control variates for inverse rendering. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 42(4), August 2023. doi: 10.1145/3592139.
- Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3D representations without 3D supervision. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3504–3515, 2020.
- Merlin Nimier-David, Sébastien Speierer, Benoît Ruiz, and Wenzel Jakob. Radiative backpropagation: An adjoint method for lightning-fast differentiable rendering. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 39(4), July 2020. doi: 10.1145/3386569.3392406.
- Jan Novák, Andrew Selle, and Wojciech Jarosz. Residual ratio tracking for estimating attenuation in participating media. *ACM Trans. Graph.*, 33(6), November 2014. ISSN 0730-0301. doi: 10.1145/2661229.2661292. URL <https://doi.org/10.1145/2661229.2661292>.
- Jan Novák, Iliyan Georgiev, Johannes Hanika, and Wojciech Jarosz. Monte Carlo methods for volumetric light transport simulation. In *Computer Graphics Forum*, volume 37, pages 551–576. Wiley Online Library, 2018.
- Art B. Owen. *Monte Carlo Theory, Methods and Examples*. Self-published, 2013. artowen.su.domains/mc.
- George C Papanicolaou. Diffusion in random media. In *Surveys in applied mathematics*, pages 205–253. Springer, 1995.
- Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. MIT Press, 2023.
- Yang Qi, Dario Seyb, Benedikt Bitterli, and Wojciech Jarosz. A bidirectional formulation for walk on spheres. In *Computer Graphics Forum*, volume 41, pages 51–62. Wiley Online Library, 2022.
- Matthias Raab, Daniel Seibert, and Alexander Keller. Unbiased global illumination with participating media. In *Monte Carlo and Quasi-Monte Carlo Methods 2006*, pages 591–605. Springer, 2006.
- Jonathan Ragan-Kelley, Connelly Barnes, Andrew Adams, Sylvain Paris, Frédo Durand, and Saman Amarasinghe. Halide: a language and compiler for optimizing parallelism, locality, and recomputation in image processing pipelines. *ACM SIGPLAN Notices*, 48(6):519–530, 2013.
- Ravi Ramamoorthi, Dhruv Mahajan, and Peter Belhumeur. A first-order analysis of lighting, shading, and shadows. *ACM Transactions on Graphics*, 26(1):2:1–2:21, 2007.
- Mani Ramanagopal and Bailey Miller. Vision, imaging, and simulation for heat and light. In *SIGGRAPH Asia 2025 Courses*, SA Courses ’25, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400721311. doi: 10.1145/3757371.3763255. URL <https://doi.org/10.1145/3757371.3763255>.

- Damien Rioux-Lavoie, Ryusuke Sugimoto, Tümay Özdemir, Naoharu H Shimada, Christopher Batty, Derek Nowrouzezahrai, and Toshiya Hachisuka. A Monte Carlo method for fluid simulation. *ACM Transactions on Graphics*, 41(6):1–16, 2022.
- Yousef Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, 2nd edition, 2003.
- Karl K. Sabelfeld and Nikolai A. Simonov. *Random walks on boundaries for solving PDES*. VSP, 1994.
- Rohan Sawhney. Walk on Spheres Extensions (WoSX), 2026.
- Rohan Sawhney and Keenan Crane. Monte Carlo geometry processing: A grid-free approach to PDE-based methods on volumetric domains. *ACM Transactions on Graphics*, 39(4), 2020.
- Rohan Sawhney, Dario Seyb, Wojciech Jarosz, and Keenan Crane. Grid-free Monte Carlo for PDEs with spatially varying coefficients. *ACM Trans. Graph.*, 41(4), July 2022. ISSN 0730-0301. doi: 10.1145/3528223.3530134. URL <https://doi.org/10.1145/3528223.3530134>.
- Rohan Sawhney, Bailey Miller, Ioannis Gkioulekas, and Keenan Crane. Walk on stars: A grid-free Monte Carlo method for PDEs with Neumann boundary conditions. *ACM Trans. Graph.*, 42(4), July 2023. ISSN 0730-0301. doi: 10.1145/3592398. URL <https://doi.org/10.1145/3592398>.
- Rohan Sawhney, Bailey Miller, Ioannis Gkioulekas, and Keenan Crane. State of the art in grid-free Monte Carlo methods for partial differential equations. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Courses*, SIGGRAPH Courses '25, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400715433. doi: 10.1145/3721241.3734001. URL <https://doi.org/10.1145/3721241.3734001>.
- Silvia Sellán and Alec Jacobson. Stochastic Poisson surface reconstruction. *ACM Transactions on Graphics*, 41(6):1–12, 2022.
- Dario Seyb, Eugene d'Eon, Benedikt Bitterli, and Wojciech Jarosz. From microfacets to participating media: A unified theory of light transport with stochastic geometry. *ACM transactions on graphics*, 43(4), 2024.
- Nicholas Sharp, Souhaib Attaiki, Keenan Crane, and Maks Ovsjanikov. DiffusionNet: Discretization agnostic learning on surfaces. *ACM Trans. Graph.*, 41(3), March 2022. ISSN 0730-0301. doi: 10.1145/3507905. URL <https://doi.org/10.1145/3507905>.
- Peter Shirley. Comments on Monte Carlo path tracing. Global Illumination mailing list, October 1996. <https://www.cs.cmu.edu/afs/cs/project/classes-ph/860.96/pub/www/montecarlo.mail>.
- Hang Si. TetGen, a Delaunay-based quality tetrahedral mesh generator. *ACM Transactions on Mathematical Software*, 41(2):11:1–11:36, 2015.
- Nikolai A Simonov. Walk-on-spheres algorithm for solving third boundary value problem. *Applied Mathematics Letters*, 64:156–161, 2017.
- Tanli Su, Bailey Miller, Bernd Bickel, Delio Vicini, and Ioannis Gkioulekas. Sobolev flows for neural field optimization. Under submission, 2026.

- Ryusuke Sugimoto. *Toward General-Purpose Monte Carlo PDE Solvers for Graphics Applications*. PhD thesis, University of Waterloo, 2025.
- Ryusuke Sugimoto, Terry Chen, Yiti Jiang, Christopher Batty, and Toshiya Hachisuka. A practical walk-on-boundary method for boundary value problems. *ACM Transactions on Graphics*, 42(4):1–16, 2023.
- Ryusuke Sugimoto, Christopher Batty, and Toshiya Hachisuka. Velocity-based Monte Carlo fluids. In *ACM SIGGRAPH 2024 Conference Papers*, 2024.
- Eric Veach. *Robust monte carlo methods for light transport simulation*. PhD thesis, Stanford, CA, USA, 1998. AAI9837162.
- Eric Veach and Leonidas J. Guibas. Optimally combining sampling techniques for Monte Carlo rendering. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, page 419–428, New York, NY, USA, 1995. Association for Computing Machinery. ISBN 0897917014. doi: 10.1145/218380.218498. URL <https://doi.org/10.1145/218380.218498>.
- Delio Vicini, Sébastien Speierer, and Wenzel Jakob. Path replay backpropagation: Differentiating light paths using constant memory and linear time. *Transactions on Graphics (Proceedings of SIGGRAPH)*, 40(4):108:1–108:14, August 2021. doi: 10.1145/3450626.3459804.
- Jiří Vorba, Ondřej Karlík, Martin Šik, Tobias Ritschel, and Jaroslav Krivánek. On-line learning of parametric mixture models for light transport simulation. *ACM Transactions on Graphics*, 33(4):1–11, 2014.
- Bruce Walter, Stephen R. Marschner, Hongsong Li, and Kenneth E. Torrance. Microfacet models for refraction through rough surfaces. In *Proceedings of the Eurographics Symposium on Rendering*, pages 195–206, 2007.
- Gregory J. Ward, Francis M. Rubinstein, and Robert D. Clear. A ray tracing solution for diffuse interreflection. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '88, page 85–92, New York, NY, USA, 1988. Association for Computing Machinery. ISBN 0897912756. doi: 10.1145/54852.378490. URL <https://doi.org/10.1145/54852.378490>.
- Ling-Qi Yan, Chi-Wei Tseng, Henrik Wann Jensen, and Ravi Ramamoorthi. Physically-accurate fur reflectance: Modeling, measurement and rendering. *ACM Transactions on Graphics*, 34(6), 2015.
- Ling-Qi Yan, Miloš Hašan, Steve Marschner, and Ravi Ramamoorthi. Position-normal distributions for efficient rendering of specular microstructure. *ACM Transactions on Graphics*, 35(4), 2016.
- Ekrem Fatih Yilmazer, Delio Vicini, and Wenzel Jakob. Solving inverse PDE problems using grid-free Monte Carlo estimators. *ACM Trans. Graph.*, 43(6), November 2024. ISSN 0730-0301. doi: 10.1145/3687990. URL <https://doi.org/10.1145/3687990>.
- Zihan Yu, Lifan Wu, Zhiqian Zhou, and Shuang Zhao. A differential Monte Carlo solver for the Poisson equation. In *ACM SIGGRAPH 2024 Conference Papers*, SIGGRAPH '24, New

- York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400705250. doi: 10.1145/3641519.3657460. URL <https://doi.org/10.1145/3641519.3657460>.
- Zihan. Yu, Rohan Sawhney, Bailey Miller, Lifan Wu, and Shuang Zhao. Robust derivative estimation with walk on stars. *ACM Trans. Graph.*, 44(6):253:1–253:16, 2025.
- Yong Zhan, Sanjay V. Kumar, and Sachin S. Sapatnekar. Thermally aware design. *Foundations and Trends® in Electronic Design Automation*, 2(3):255–370, 2008. ISSN 1551-3939. doi: 10.1561/10000000007.
- Cheng Zhang, Lifan Wu, Changxi Zheng, Ioannis Gkioulekas, Ravi Ramamoorthi, and Shuang Zhao. A differential theory of radiative transfer. *ACM Trans. Graph.*, 38(6):227:1–227:16, 2019.
- Cheng Zhang, Bailey Miller, Kai Yan, Ioannis Gkioulekas, and Shuang Zhao. Path-space differentiable rendering. *ACM Trans. Graph.*, 39(4):143:1–143:19, 2020.
- Xian Yao Zhang, Gerhard R  thlin, Shilin Zhu, Tun   Ozan Aydın, Farnood Salehi, Markus Gross, and Marios Papas. Neural denoising for deep-Z Monte Carlo renderings. *Computer Graphics Forum (Proc. Eurographics)*, 2024.
- Shuang Zhao, Milo   Ha  san, Ravi Ramamoorthi, and Kavita Bala. Modular flux transfer: Efficient rendering of high-resolution volumes with repeated structures. *ACM Transactions on Graphics*, 32(4):1–12, 2013.
- Shuang Zhao, Fr  do Durand, and Changxi Zheng. Inverse diffusion curves using shape optimization. *IEEE transactions on visualization and computer graphics*, 24(7):2153–2166, 2017.
- Yijing Zhou, Wei Cai, and Elton Hsu. Computation of local time of reflecting Brownian motion and probabilistic representation of the Neumann problem. *arXiv preprint arXiv:1502.01319*, 2015.
- Zihong Zhou, Eugene d’Eon, Rohan Sawhney, and Wojciech Jarosz. Harmonic caching for walk on spheres. *ACM Transactions on Graphics*, 44(6):1–15, 2025.