Robust Derivative Estimation with Walk on Stars

ZIHAN YU, University of California, Irvine, USA
ROHAN SAWHNEY, NVIDIA, USA
BAILEY MILLER, Carnegie Mellon University, USA
LIFAN WU, NVIDIA, USA
SHUANG ZHAO, University of Illinois Urbana-Champaign, USA

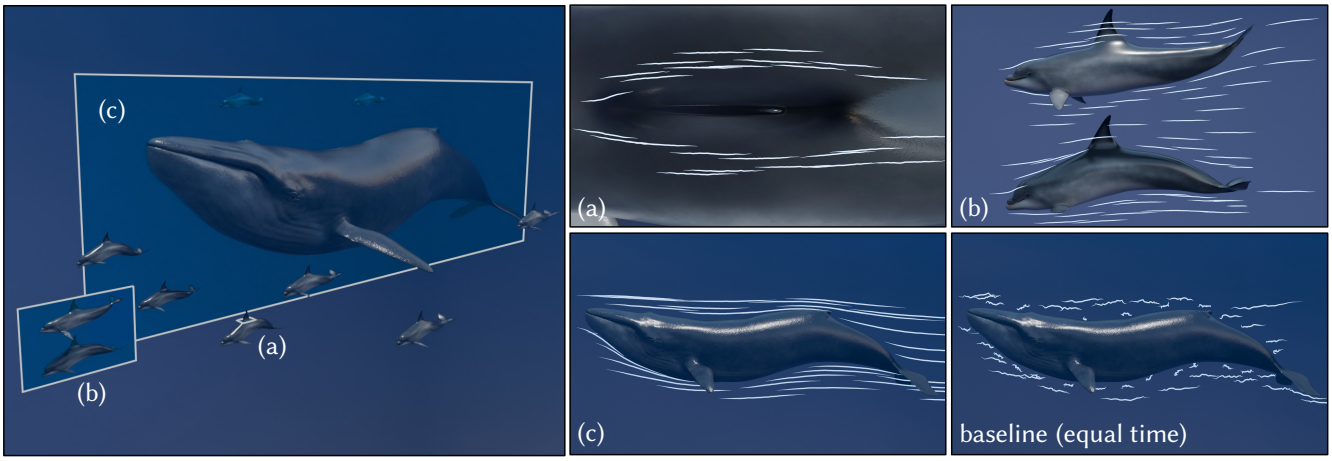


Fig. 1. Streamlines from a potential flow simulation around marine life of vastly different scales, computed using our Monte Carlo walk on stars solver for spatial derivatives. Unlike traditional solvers, our method can compute flow gradients at *arbitrary* resolutions for streamline tracing in local regions of interest–whether around a single fin (a), multiple dolphins (b), or a full blue whale (c)–without requiring a background grid or a volumetric mesh adapted to the boundary geometry. Compared to prior walk on stars estimators (*bottom right*), our method achieves significantly lower error at equal computation time.

Monte Carlo methods based on the walk on spheres (WoS) algorithm offer a parallel, progressive, and output-sensitive approach for solving partial differential equations (PDEs) in complex geometric domains. Building on this foundation, the walk on stars (WoSt) method generalizes WoS to support mixed Dirichlet, Neumann, and Robin boundary conditions. However, accurately computing spatial derivatives of PDE solutions remains a major challenge: existing methods exhibit high variance and bias near the domain boundary, especially in Neumann-dominated problems. We address this limitation with a new extension of WoSt specifically designed for derivative estimation. Our method reformulates the boundary integral equation (BIE) for Poisson PDEs by directly leveraging the harmonicity of spatial derivatives. Combined with a tailored random-walk sampling scheme and an unbiased early termination strategy, we achieve significantly improved accuracy in derivative estimates near the Neumann boundary. We further demonstrate the effectiveness of our approach across various tasks, including recovering the non-unique solution to a pure Neumann problem with reduced bias and variance, constructing divergence-free vector fields, and optimizing parametrically defined boundaries under PDE constraints.

CCS Concepts: • **Computing methodologies** → **Physical simulation**.

Additional Key Words and Phrases: Monte Carlo methods, partial differential equations, walk on spheres, differentiable simulation

ACM Reference Format:
Zihan Yu, Rohan Sawhney, Bailey Miller, Lifan Wu, and Shuang Zhao. 2025. Robust Derivative Estimation with Walk on Stars. *ACM Trans. Graph.* 44, 6, Article 253 (December 2025), 16 pages. https://doi.org/10.1145/3763333

1 INTRODUCTION

Recent years have seen significant advances in Monte Carlo solvers for partial differential equations within the computer graphics community, particularly those built on Muller's walk on spheres (WoS) algorithm [Muller 1956]. WoS offers a compelling alternative to grid-based methods for solving fundamental PDEs such as the Laplace equation, especially in geometrically complex domains [Sawhney and Crane 2020]. Much like Monte Carlo path tracing [Pharr et al. 2023], WoS-based solvers embrace randomness to gain key numerical advantages such as output-sensitive computation, natural parallelism, and robustness to complex geometry, all without requiring background grids or the inversion of large linear systems.

Yet a significant limitation remains: current Monte Carlo PDE solvers are not well-equipped to estimate spatial derivatives of solutions. Gradients are essential for analyzing how solutions vary

Authors' addresses: Zihan Yu, zihay19@uci.edu, University of California, Irvine, USA; Rohan Sawhney, rsawhney@nvidia.com, NVIDIA, USA; Bailey Miller, bmmiller@andrew.cmu.edu, Carnegie Mellon University, USA; Lifan Wu, lifanw@nvidia.com, NVIDIA, USA; Shuang Zhao, shzhao@illinois.edu, University of Illinois Urbana-Champaign, USA.

ACM 0730-0301/2025/12-ART253
https://doi.org/10.1145/3763333

across space, with applications in computing heat flux, voltage and pressure drops [Sawhney et al. 2023; Bati et al. 2023; Miller et al. 2024b], as well as performing gradient-based optimization of shapes and materials [Yu et al. 2024; Miller et al. 2024a; Yilmazer et al. 2024]. Unfortunately, derivative estimates are typically much noisier than solution estimates, especially near boundaries, because the estimator involves a singular kernel. This issue is exacerbated in boundary value problems (BVPs) dominated by Neumann conditions, where estimation requires evaluating high-dimensional integrals via long random walks that repeatedly reflect off the Neumann boundary and can only terminate on the Dirichlet boundary (Figure 2).

To address these challenges, we extend the walk on stars algorithm [Sawhney et al. 2023; Miller et al. 2024b]–a generalization of WoS for solving Poisson equations with mixed Dirichlet, Neumann, and Robin boundary conditions–to enable more robust spatial derivative estimation. Our method achieves significantly reduced error compared to existing WoSt-based derivative estimators. Specifically, the estimators we develop (Section 4):

- maintain bounded variance both near the boundary and within the domain by carefully handling singular kernels and incorporating control variates;
- reduce variance in Neumann-dominated problems through early, unbiased termination of random walks, avoiding the longer trajectories required to estimate the solution itself;
- preserve key strengths of Monte Carlo PDE solvers, such as output sensitivity, and parallel and progressive computation.

Beyond improving derivative estimation, our method also unlocks new capabilities that were previously out of reach for Monte Carlo solvers, such as:

- A tractable estimator for the non-unique *solution* to pure Neumann problems, offering substantially lower bias and variance than prior approaches (Section 6.2).
- Gradient-based optimization of parametrically defined Neumann boundaries (Section 6.4). Previous Monte Carlo methods have largely avoided these problems, since computing parameter gradients requires estimating second-order spatial derivatives involving hypersingular kernels.

At the core of our method is a key mathematical insight: the derivative of a harmonic function is itself harmonic. This property enables us to formulate boundary integral equations for first- and second-order spatial derivatives of Poisson equations, which we estimate using the walk on stars framework.

While our method addresses key challenges in derivative estimation, it assumes a smooth boundary and does not handle singular behavior at sharp concave corners. Future work includes extending the approach to non-smooth geometries, incorporating Robin boundary conditions, and generalizing beyond Poisson equations.

2 RELATED WORK

We provide an overview of the Monte Carlo PDE solvers directly relevant to our method, with a focus on approaches for estimating spatial derivatives. For a broader introduction to Monte Carlo solvers for PDEs and a discussion of their numerical tradeoffs relative to grid-based methods, we refer readers to resources by Sawhney et al. [2025] and Sawhney [2024].

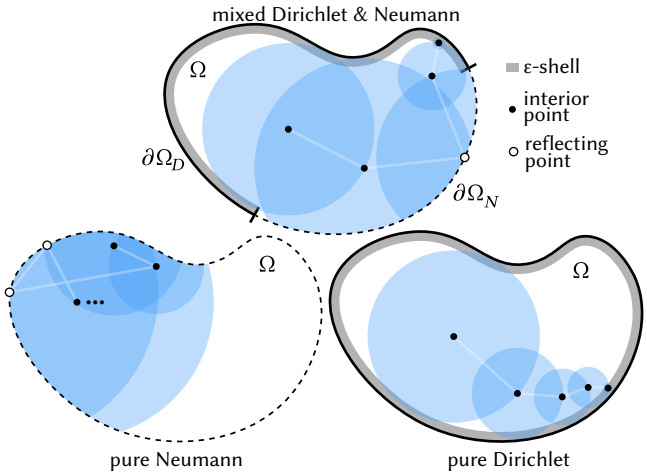


Fig. 2. Walk on stars solves Poisson equations with mixed Dirichlet and Neumann boundary conditions by taking independent random walks starting from any given point in the domain (*top*). Walks reflect off the Neumann boundary and terminate on the Dirichlet boundary. As a result, WoSt does not terminate in pure Neumann problems (*bottom left*), and reduces to Muller's walk on spheres in pure Dirichlet problems (*bottom right*).

2.1 Monte Carlo Solvers for Partial Differential Equations

Since its introduction to the graphics community [Sawhney and Crane 2020], the walk on spheres algorithm [Muller 1956] has been extended far beyond its original application to Laplace equations with Dirichlet boundary conditions. Recent work has broadened its applicability to a wider class of linear elliptic equations [Sawhney et al. 2022; De Lambilly et al. 2023; Sugimoto et al. 2024b; Miller et al. 2025] and to more general boundary conditions [Sawhney et al. 2023; Miller et al. 2024b; Sugimoto et al. 2023], along with the development of several advanced sampling and variance reduction strategies [Nabizadeh et al. 2021; Qi et al. 2022; Miller et al. 2023; Bakbouk and Peers 2023; Li et al. 2023, 2024; Huang et al. 2025]. Together, these advancements are now enabling Monte Carlo PDE solvers to be applied in diverse settings, including heat transfer [Bati et al. 2023], geometry processing [Sawhney and Crane 2020; de Goes and Desbrun 2024], fluid simulation [Rioux-Lavoie et al. 2022; Jain et al. 2024; Sugimoto et al. 2024a], differentiable rendering [Wu et al. 2025], and inverse geometric optimization [Yu et al. 2024; Miller et al. 2024a; Yilmazer et al. 2024].

Our approach to computing spatial derivatives builds on the walk on stars method, which solves boundary value problems with arbitrary first-order linear boundary conditions [Sawhney et al. 2023; Miller et al. 2024b]. Alternative methods such as walk on boundary (WoB) [Sugimoto et al. 2023] also solve similar BVPs, but WoB exhibits a significantly less favorable bias-variance tradeoff than WoSt in non-convex domains for both solutions and derivatives [Miller et al. 2024b, Figure 12 & Table 1]. Consequently, our method inherits the advantages of WoSt over WoB for derivative estimation.

2.2 Derivative Estimation with Monte Carlo Solvers

Sawhney and Crane [2020, Section 3] introduced a mean-value integral over a ball to evaluate the spatial gradient of a Poisson equation, which can be estimated using both WoS and WoSt (Section 3.3). However, this formulation requires a non-zero ball radius and thus cannot be applied directly at the domain boundary. Moreover, the integral involves a singular kernel that diverges as the ball radius shrinks to zero—*e.g.*, when a random walk approaches the boundary. This singularity persists even under gradient-specific variance reduction techniques such as control variates and antithetic sampling [Sawhney and Crane 2020; Rioux-Lavoie et al. 2022].

In Neumann-dominated BVPs, noise in WoSt-based solution and gradient estimators is further amplified by the recursive sampling of high-dimensional integrals arising from reflected random walks (Figure 2). Boundary value caching (BVC) [Miller et al. 2023, 2024b] seeks to amortize this cost by initiating walks only from the boundary and reusing their results to estimate values in the interior. While BVC reduces variance through correlated sampling away from the boundary, its gradient estimates near the boundary exhibit even greater noise than those of WoSt due to hypersingular kernels. In some cases, the estimators may even fail to converge. Furthermore, since BVC relies on WoSt for generating random walks, it inherits the same challenges in Neumann dominated problems: namely, the need to simulate long, high-variance walks that terminate only on the Dirichlet boundary, or to apply ad hoc termination strategies such as *Tikhonov regularization* for pure Neumann problems, which introduce non-negligible bias [Sawhney et al. 2023, Section 6.4].

Our method mitigates the effects of singular kernels by modifying the *star-shaped regions* used in WoSt to perform random walks. In particular, as in the WoSt formulation for Robin boundary conditions (Section 3.2), we introduce a *reflectance* function into our BIE for first-order spatial derivatives (Equation 9). This function governs the radius of each star-shaped region, allowing us to estimate derivatives directly on the boundary or in the interior, while also ensuring bounded variance. It also enables unbiased early termination of walks via Russian roulette. Together, these modifications make our method significantly more efficient at computing derivatives in Neumann-dominated problems, including pure Neumann cases.

Finally, Yu et al. [2024] recently proposed using an off-centered sphere at the boundary to mitigate kernel singularities, but their method is limited to computing normal derivatives on the Dirichlet boundary. In Section 6.4, we extend this approach to compute second-order normal derivatives on the Neumann boundary, enabling optimization of parametrically defined geometry under Neumann constraints. In contrast, prior work on differentiable Monte Carlo PDE solvers [Yılmazer et al. 2022; Yu et al. 2024; Miller et al. 2024a] does not address Neumann boundary optimization, and instead computes parameter derivatives either for material coefficients inside the domain or Dirichlet conditions on the boundary.

3 BACKGROUND

We review the key concepts and techniques underlying our method for computing spatial and parameter derivatives of Poisson PDEs, namely boundary integral equations and Monte Carlo estimation using the walk on stars algorithm.

Notation. Let $\phi(x)$ be a function defined on a domain $\Omega \subset \mathbb{R}^3$ with boundary $\partial\Omega$. We use $\nabla\phi$ to denote its spatial gradient, given by $(\partial_x\phi, \partial_y\phi, \partial_z\phi)$. For any vector $\boldsymbol{v} \in \mathbb{R}^3$, the *directional derivative* of ϕ along $\boldsymbol{v}$ is denoted by $\partial_v\phi := \boldsymbol{v} \cdot \nabla\phi$. On $\partial\Omega$, we denote the normal derivative by $\partial_n\phi$, and the tangential gradient by $\nabla_\Gamma\phi$. If the domain $\Omega(\pi)$ is parameterized by a finite-dimensional vector $\pi \in \mathbb{R}^N$, we write $\dot{\phi}$ for the *parameter derivative* of a π-dependent function $\phi(x, \pi)$, *i.e.*, the partial derivative with respect to the parameters π.

3.1 Boundary Integral Formulation For Poisson Equation

We consider PDEs of the form

$$\begin{aligned} \Delta u &= -f && \text{on } \Omega, \\ u &= g && \text{on } \partial\Omega_{\mathrm{D}}, \\ \partial_n u &= h && \text{on } \partial\Omega_{\mathrm{N}}, \end{aligned} \tag{1}$$

where Δ is the negative-semidefinite Laplacian, and $f : \Omega \to \mathbb{R}$ is the source term. The boundary $\partial\Omega$ is partitioned into a Dirichlet part $\partial\Omega_{\mathrm{D}}$ with prescribed values $g : \partial\Omega_{\mathrm{D}} \to \mathbb{R}$, and a Neumann part $\partial\Omega_{\mathrm{N}}$ with prescribed normal derivatives $h : \partial\Omega_{\mathrm{N}} \to \mathbb{R}$. Assuming $\partial\Omega$ is smooth, the solution satisfies the boundary integral equation [Costabel 1987; Hunter and Pullan 2001]

$$\begin{aligned} \alpha(x)\, u(x) = &\int_{\partial \mathrm{A}} P^{\mathrm{C}}(x,z)\, u(z) \;-\; G^{\mathrm{C}}(x,z)\, \partial_n u(z)\, \mathrm{d}z \\ &+ \int_{\mathrm{A}} G^{\mathrm{C}}(x,y)\, f(y)\, \mathrm{d}y \end{aligned} \tag{2}$$

for any point $x \in \mathbb{R}^3$. Here A and C are arbitrary subsets of Ω and $\mathbb{R}^3$ (respectively), and $\alpha(x) = 1$ if $x \in A$, $1/2$ if $x \in \partial A$, and 0 otherwise. Explicit expressions for the *Green's function* G^{C} and *Poisson kernel* P^{C} are known in free-space ($\mathrm{C} = \mathbb{R}^3$) and for balls ($\mathrm{C} = \mathrm{B}(x, R)$ with radius R) [Sawhney 2024, Appendix A]. To evaluate this BIE at a point x, one must know u and $\partial_n u$ at all points $z \in \partial\mathrm{A}$. However these quantities are often only partially specified by the boundary conditions g on $\partial\Omega_{\mathrm{D}}$ and h on $\partial\Omega_{\mathrm{N}}$.

Moreover, for a given direction $\boldsymbol{v}$, one can show–via implicit differentiation [Henrot and Pierre 2018, Section 5.5]–that the directional derivative $\partial_v u$ of Equation 1 also satisfies a Poisson equation. As a consequence, $\partial_v u(x)$ admits a BIE similar in form to Equation 2 with unknowns $\partial_v u(z)$ and $\partial^2_{nv} u(z)$. We make use of such a BIE in Section 4 to develop our method for estimating spatial derivatives.

3.2 Walk on Stars

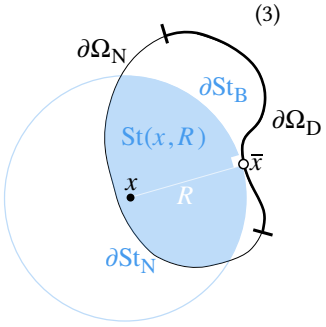
The walk on stars method [Sawhney et al. 2023; Miller et al. 2024b] solves a Poisson equation at any point $x_0 \in \Omega$ by performing independent random walks within the domain (Figure 2). Each walk accumulates contributions from the source term f in the interior and from the Neumann boundary data h when reflecting off $\partial\Omega_{\mathrm{N}}$. Walks terminate upon reaching the Dirichlet boundary, defined as being within an ε distance of $\partial\Omega_{\mathrm{D}}$. At termination, they collect the value g from the closest projected point $\bar{x}_k \in \partial\Omega_{\mathrm{D}}$ to the final walk location x_k (for $k \geq 0$).

3.2.1 Boundary Integral Formulation. Concretely, WoSt acts as a Monte Carlo estimator for the boundary integral in Equation 2 by selecting A to be a *star-shaped region* $\mathrm{St}(x, R)$ relative to the current random walk location x (inset). This region is constructed by intersecting the domain Ω with a ball $\mathrm{B}(x, R)$, where the radius

R is chosen as the minimum distance from x to the closest *silhouette* point on $\partial\Omega_N$, and to the closest point on $\partial\Omega_D$. With this choice of A, the BIE involves only a single unknown function $u(z)$:

$$\alpha(x)\,u(x) = \int_{\partial \mathrm{St}} P^{\mathrm{B}}(x,z)\,u(z)\,\mathrm{d}z \;-\; \int_{\partial \mathrm{St}_N} G^{\mathrm{B}}(x,z)\,h(z)\,\mathrm{d}z + \int_{\mathrm{St}} G^{\mathrm{B}}(x,y)\,f(y)\,\mathrm{d}y. \tag{3}$$

Here, $\partial\mathrm{St}_N$ denotes the portion of the star-shaped boundary $\partial\mathrm{St}$ that lies on the Neumann boundary, where the normal derivative $\partial_n u(z) = h$ is prescribed. On the spherical portion $\partial\mathrm{St}_B \coloneqq \partial\mathrm{B} \cap \partial\mathrm{St}$, the Green's function G^{B} vanishes, so $\partial_n u(z)$ does not contribute and need not be evaluated.



3.2.2 Monte Carlo Estimation. The integrals involving the Neumann boundary data h and the source term f in Equation 3 contain no unknowns and can be estimated directly. We refer readers to Sawhney et al. [2023, Sections 4.5 & 4.6] for Monte Carlo strategies used to evaluate these terms. For the remaining integral involving the unknown function $u(z)$, WoSt employs a *single-sample* Monte Carlo estimator at each step $x_k \in \Omega$ of the random walk:

$$\widehat{u}(x_k) = \frac{P^{\mathrm{B}}(x_k, x_{k+1})\,\widehat{u}(x_{k+1})}{\alpha(x_k)\,p^{\partial\mathrm{St}(x_k,R)}(x_{k+1})}. \tag{4}$$

This estimator is *recursive*, as $\widehat{u}$ appears on both sides of the equation. The next walk location x_{k+1} is sampled from the probability density function $p^{\partial\mathrm{St}}$ defined over $\partial\mathrm{St}(x_k, R)$.

Conveniently, the Poisson kernel $P^{\mathrm{B}}(x_k, x_{k+1})$ is the signed solid angle subtended by $\partial\mathrm{St}$ at x_k [Sawhney et al. 2023, Equation 25]. Thus, similar to Monte Carlo path tracing, WoSt uses *direction sampling* to determine x_{k+1}: it casts a ray from x_k in a direction uniformly sampled from the unit sphere and takes the first intersection with $\partial\mathrm{St}$. If x_k lies on the Neumann boundary $\partial\Omega_N$, the ray direction is instead sampled from a hemisphere aligned with the inward normal, ensuring the walk remains inside the domain. Under this sampling scheme, the ratio $P^{\mathrm{B}}(x_k, x_{k+1})/[\alpha(x_k)p^{\partial\mathrm{St}}(x_{k+1})]$ becomes 1 at each step, leaving $\widehat{u}$ unchanged from the multiplicative identity.

3.2.3 Pure Neumann conditions. In the absence of a Dirichlet boundary, random walks under WoSt never terminate (Figure 2, *bottom left*) and continue accumulating contributions from f and h indefinitely. This behavior reflects the underlying structure of pure Neumann problems, in which solutions are defined only up to an additive constant. Terminating walks arbitrarily–such as by imposing a maximum walk length–introduces bias that depends on the chosen termination criterion.

Sawhney et al. [2023] propose addressing this issue via Tikhonov regularization [Tikhonov 1998], which replaces the original Poisson equation with a *screened* version incorporating an adaptively set absorption coefficient σ. This modification allows walks to be terminated stochastically within the domain, with the likelihood of absorption increasing with σ. While this approach reduces bias and variance relative to fixed-length truncation, selecting an appropriate σ remains challenging: larger values induce greater bias, while smaller values lead to higher variance caused by longer walks. In Section 6.2, we demonstrate that our method can be used to compute the solution to pure Neumann problems up to an additive constant, with significantly less error and walks of finite length.

3.2.4 The Reflectance Function. For partially absorbing and reflecting Robin boundary conditions of the form $\partial_n u + \mu u = \ell$ with $\mu > 0$, Miller et al. [2024b] generalize the WoSt estimator in Equation 4 by introducing a *reflectance function*,

$$\rho(x_k, x_{k+1}) \coloneqq \begin{cases} 1, & \text{on } \partial\mathrm{St}_B, \\ 1 - \mu(x_{k+1})\,\dfrac{G^{\mathrm{B}}(x_k,x_{k+1})}{P^{\mathrm{B}}(x_k,x_{k+1})}, & \text{on } \partial\mathrm{St}_R, \end{cases} \tag{5}$$

where $\partial\mathrm{St}_R$ denotes the portion of $\partial\mathrm{St}$ on the Robin boundary $\partial\Omega_R$. At each step k, the running estimate $\widehat{u}$ is multiplied by the reflectance $\rho(x_k, x_{k+1})$. Of particular importance to our method, ρ informs the selection of the radius R for the star-shaped region St, so that reflectance remains within the range $[0, 1]$. This boundedness allows ρ to also serve as a survival probability in a Russian roulette scheme, enabling unbiased early termination of walks on $\partial\Omega_R$. In Section 4, we develop a modified WoSt estimator for spatial derivatives of Equation 1, which similarly uses a reflectance function to terminate walks on the Neumann boundary without introducing bias.

3.3 Estimating Spatial Derivatives with Walk on Stars

For a ball $\mathrm{B}(c, R)$ centered at a point $c \in \Omega$ and contained within the domain, Yu et al. [2024], building on Sawhney and Crane [2020]; Sawhney et al. [2022], use the following integral expression to evaluate the spatial gradient ∇u of a Poisson equation at a point x:

$$\nabla u(x) = \int_{\partial \mathrm{B}(c,R)} \nabla P^{\mathrm{B}}(x,z) u(z)\,\mathrm{d}z \;+\; \int_{\mathrm{B}(c,R)} \nabla G^{\mathrm{B}}(x,y) f(y)\,\mathrm{d}y. \tag{6}$$

This formulation is convenient because unknown values $u(z)$ can be estimated recursively using WoSt. However, sampling z uniformly on the sphere $\partial\mathrm{B}$ does not properly account for the radial singularity in the kernel ∇P^{B}. This singularity becomes increasingly ill-behaved as c approaches the boundary and R reduces to 0, resulting in noisy estimates near the boundary and incorrect estimates directly on it.

To address this issue, Sawhney and Crane [2020, Section 4] and Rioux-Lavoie et al. [2022, Section 4] propose control and antithetic variate strategies, respectively, for the case where the evaluation point x coincides with the center c in Equation 6. While these techniques reduce noise near the boundary, the singularity in ∇P^{B} persists. We compare our method with this baseline in Section 6.1.

Sawhney and Crane [2020, Section 3] also provide an expression for the Hessian of u, but it involves hypersingular kernels that result in even higher variance. In Section 4.4, we introduce a more tractable estimator for the second normal derivative $\partial_n^2 u$.

3.4 PDE-Constrained Shape Optimization

Inverse problems involving the optimization of domain shape under PDE constraints arise across science and engineering, from airfoil and heat-sink design [Hicks and Henne 1977; Zhan et al. 2008] to structural lightweighting [Allaire et al. 2014]. While Monte

Carlo solvers have recently been applied to inverse problems involving Poisson-like PDEs [Yu et al. 2024; Miller et al. 2024a; Yilmazer et al. 2024], prior work does not support parameterized Neumann boundary conditions $h(x, \pi)$. This omission stems from the fact that optimizing arbitrary parameters π requires solving a differential version of Equation 1 [Henrot and Pierre 2018, Equation 5.79]:

$$\begin{aligned} \Delta \dot{u} &= 0 && \text{on } \Omega, \\ \dot{u} &= 0 && \text{on } \partial\Omega_{\mathrm{D}}, \\ \partial_n \dot{u} &= \left(\partial_n h - \partial_n^2 u\right) \mathrm{V}_n + \nabla u \cdot \nabla_\Gamma \mathrm{V}_n && \text{on } \partial\Omega_{\mathrm{N}}, \end{aligned} \tag{7}$$

where $\dot{u}$ is the parameter derivative and V_n denotes the normal velocity of the boundary. For simplicity, we assume that the source term f and Dirichlet data g do not depend on π; prior work addresses these cases. Directly solving this PDE with WoSt is challenging due to the *nested* dependence of $\dot{u}$ on first and second-order spatial derivatives–specifically ∇u and $\partial_n^2 u$–for which existing Monte Carlo estimators can be inefficient. Our method addresses this challenge by providing more reliable estimates of these derivative terms.

4 METHOD

In this section, we develop an alternative to the baseline WoSt estimator for spatial derivatives described in Section 3.3, which does not suffer from singular kernels and can be evaluated directly on the domain boundary. We begin by deriving a boundary integral equation in Section 4.1 that is specifically tailored towards estimating spatial derivatives. A key advantage of this formulation–detailed in Section 4.2–is that unlike random walks for the solution estimator which must always reflect off the Neumann boundary, walks for the derivative estimator can be terminated early without introducing bias. This leads to shorter walk lengths and, as we show in Section 6, significantly lower error compared to the baseline approach.

We focus initially on pure Neumann problems in Sections 4.1 and 4.2 to describe the core components of our estimator. We then extend our method to mixed Dirichlet–Neumann problems in Section 4.3. In Section 4.4 we introduce an estimator for the second normal derivative of a Poisson equation, which we use for gradient-based optimization of Neumann boundaries in Section 6.4. Section 5 then provides implementation details for triangle meshes.

4.1 Boundary Integral Equation for Directional Derivatives

As discussed in Section 3.1, our method builds on the insight that just like the solution u to a Poisson equation, its directional derivative $\partial_v u$ along any direction $\boldsymbol{v}$ also satisfies a BIE over a star-shaped region St. In particular, we have

$$\alpha(x)\, \partial_v u(x) = \int_{\partial \mathrm{St}} P^{\mathrm{B}}(x,z)\, \partial_v u(z)\, \mathrm{d}z - \int_{\partial \mathrm{St}_{\mathrm{N}}} G^{\mathrm{B}}(x,z)\, \partial_v^2 u(z)\, \mathrm{d}z + \int_{\mathrm{St}} G^{\mathrm{B}}(x,y)\, \partial_v f(y)\, \mathrm{d}y. \tag{8}$$

However, unlike the boundary integral in Equation 3, a key challenge in estimating Equation 8 is the presence of *two* unknown functions: the directional derivative $\partial_v u(z)$ and the second-order mixed derivative $\partial_{nv}^2 u(z)$. Estimating both terms simultaneously using WoSt would require a branching random walk, where new walks would need to be spawned to estimate each nested derivative. This would lead to significantly higher computational and memory costs per sample. Moreover, as the boundary conditions in Equation 1 are specified only for the solution u and its normal derivative, the boundary constraints governing $\partial_v u$ and $\partial_{nv}^2 u$ are not well defined.

To overcome these issues, we first reformulate the boundary integral above to follow a structure parallel to Equation 3, involving only a single unknown function. In Appendix A, we show that this transformation can be achieved by applying certain vector calculus identities to the second-order derivative term $\partial_{nv}^2 u$, and performing integration by parts on the second integral in Equation 8. Assuming $\partial\Omega_{\mathrm{N}}$ is smooth, the resulting integral expression is

$$\begin{aligned} \alpha(x)\, \partial_v u(x) = &\int_{\partial \mathrm{St}} P^{\mathrm{B}}(x,z)\, \left(|\boldsymbol{\rho}_v(x,z)| \partial_\rho u(z) + \mu_v(z)\right)\, \mathrm{d}z \\ &- \int_{\partial \mathrm{St}_{\mathrm{N}}} G^{\mathrm{B}}(x,z)\, \eta_v(z)\, \mathrm{d}z \\ &+ \int_{\mathrm{St}} G^{\mathrm{B}}(x,y)\, \partial_v f(y)\, \mathrm{d}y, \end{aligned} \tag{9}$$

where the auxiliary functions $\boldsymbol{\rho}_v$, μ_v and η_v are defined as

$$\boldsymbol{\rho}_v(x,z) = \begin{cases} \boldsymbol{v} & \text{if } z \in \partial \mathrm{St}_{\mathrm{B}} \\ \boldsymbol{v}_\Gamma - \boldsymbol{v} \cdot \boldsymbol{n} \frac{\nabla_\Gamma G^{\mathrm{B}}(x,z)}{P^{\mathrm{B}}(x,z)} & \text{if } z \in \partial \mathrm{St}_{\mathrm{N}}, \end{cases} \tag{10}$$

$$\mu_v(z) = \begin{cases} 0 & \text{if } z \in \partial \mathrm{St}_{\mathrm{B}} \\ \boldsymbol{v} \cdot \boldsymbol{n} h(z) & \text{if } z \in \partial \mathrm{St}_{\mathrm{N}}, \end{cases} \tag{11}$$

$$\eta_v(z) = \partial_{v_\Gamma} h(z) - \boldsymbol{v} \cdot \boldsymbol{n} H(z) h(z) - \boldsymbol{v} \cdot \boldsymbol{n} f(z). \tag{12}$$

Here, $\boldsymbol{v}_\Gamma$ denotes the tangential components of the input direction $\boldsymbol{v}$ along the Neumann boundary with outward normal $\boldsymbol{n}$, and H is the boundary's mean curvature. We term $\boldsymbol{\rho}_v$ the *reflectance function*, for reasons detailed in Section 4.2. Unlike its scalar counterpart for Robin boundary conditions (Section 3.2.4), $\boldsymbol{\rho}_v$ is vector-valued in this setting. Accordingly, we use $\partial_\rho u(z)$ in Equation 9 as a shorthand for the directional derivative of u evaluated along the direction of $\boldsymbol{\rho}_v(x,z)$, as defined by the two cases in Equation 10.

Monte Carlo Estimation. All terms in Equation 9 other than $\partial_\rho u$ depend solely on known boundary data, geometric quantities, and the input direction $\boldsymbol{v}$. This structure enables the construction of a non-branching WoSt estimator for the directional derivative. Other spatial derivatives, such as the gradient, divergence and curl, can be assembled by evaluating directional derivatives along appropriate coordinate axes or field-aligned directions.

We can estimate the second and third integrals in Equation 9, involving η_v and $\partial_v f$ respectively, using the same Monte Carlo strategies developed for the corresponding terms in the original WoSt solution estimator (Section 3.2). We refer readers to Sawhney et al. [2023, Sections 4.5 & 4.6] for details. For the first integral, we follow the approach in Section 3.2.2 to formulate a recursive single-sample estimator at each step $x_k \in \Omega$ of a random walk:

$$\widehat{\partial_v u}(x_k) = \frac{P^{\mathrm{B}}(x_k, x_{k+1})\, \left(|\boldsymbol{\rho}_v(x_k, x_{k+1})| \widehat{\partial_\rho u}(x_{k+1}) + \mu_v(x_{k+1})\right)}{\alpha(x_k)\, p^{\partial \mathrm{St}(x_k, R)}(x_{k+1})}. \tag{13}$$

As before, the next walk location x_{k+1} is determined using direction sampling. As a result, the running estimate on the right hand side is updated using a multiplicative factor $|\boldsymbol{\rho}_v|$ and a known additive

ALGORITHM 1: ∂WalkOnStars$(x,\ \boldsymbol{n},\ \boldsymbol{v},\ \varepsilon)$

Input: Starting location $x \in \Omega$ of random walk, normal $\boldsymbol{n}$ at x (undefined if $x \notin \partial\Omega_{\mathrm{N}}$), direction $\boldsymbol{v}$ for derivative, ε-shell.
Output: Single-sample directional derivative estimate $\widehat{\partial_{\boldsymbol{v}} u}(x)$ for a Poisson equation with pure Neumann boundary conditions.

1: $R \leftarrow$ ComputeStarRegionRadius$(x,\ \boldsymbol{v})$ ▷ *Compute radius of star region* $\mathrm{St}(x, R)$ *containing Neumann boundary* $\partial\Omega_{\mathrm{N}}$ *(Sections 5.1-5.2)*
2: $R \leftarrow \max(\varepsilon,\ R)$ ▷ *Ensure* $R \geq \varepsilon$ *to prevent walk from stalling on concave part of* $\partial\Omega_{\mathrm{N}}$ *[Sawhney et al. 2023, Figure 9]*
3: $\boldsymbol{d} \leftarrow$ SampleUnitSphere() ▷ *Sample a direction* $\boldsymbol{d}$ *uniformly on the unit sphere*
4: **if** $x \in \partial\Omega_{\mathrm{N}}$ **and** $\boldsymbol{n} \cdot \boldsymbol{d} > 0$ **then** $\boldsymbol{d} \leftarrow -\boldsymbol{d}$ ▷ *If* x *lies on* $\partial\Omega_{\mathrm{N}}$*, ensure* $\boldsymbol{d}$ *is sampled on hemisphere with axis* $-\boldsymbol{n}$
5: hit, p, $\boldsymbol{n} \leftarrow$ IntersectNeumannBoundary$(x,\ \boldsymbol{d},\ R)$ ▷ *Intersect* $\partial\mathrm{St}_{\mathrm{N}}$ *with ray* $x + R\,\boldsymbol{d}$*, and get first hit*
6: **if not** hit **then** $p \leftarrow x + R\,\boldsymbol{d}$ ▷ *If there is no hit with* $\partial\mathrm{St}_{\mathrm{N}}$*, update next walk location to point on* $\partial\mathrm{St}_{\mathrm{B}}$ *instead*
7: $\widehat{I}_\eta \leftarrow$ NeumannBoundaryEstimate$(x,\ \boldsymbol{v},\ R)$ ▷ *Estimate boundary contribution on* $\partial\mathrm{St}_{\mathrm{N}}$ *(second integral in Equation 9)*
8: $\widehat{I}_{\partial_{\boldsymbol{v}} f} \leftarrow$ SourceEstimate$(x,\ p,\ \boldsymbol{d},\ R)$ ▷ *Estimate source contribution in* St *(third integral in Equation 9)*
9: $\boldsymbol{\rho}_{\boldsymbol{v}} \leftarrow$ hit ? $\boldsymbol{v} - \boldsymbol{v} \cdot \boldsymbol{n} \frac{p-x}{(p-x)\cdot\boldsymbol{n}}$: $\boldsymbol{v}$ ▷ *Compute reflectance (Equation 14)*
10: **if** $|\boldsymbol{\rho}_{\boldsymbol{v}}| <$ SampleUniform$(0, 1)$ **then return** $\mu_{\boldsymbol{v}}(x,\ p,\ \boldsymbol{n},\ \boldsymbol{v})\ -\ \widehat{I}_\eta\ +\ \widehat{I}_{\partial_{\boldsymbol{v}} f}$ ▷ *Probabilistically terminate walk using Russian roulette*
11: **return** ∂WalkOnStars$(p,\ \boldsymbol{n},\ \boldsymbol{\rho}_{\boldsymbol{v}}/|\boldsymbol{\rho}_{\boldsymbol{v}}|,\ \varepsilon)\ +\ \mu_{\boldsymbol{v}}(x,\ p,\ \boldsymbol{n},\ \boldsymbol{v})\ -\ \widehat{I}_\eta\ +\ \widehat{I}_{\partial_{\boldsymbol{v}} f}$ ▷ *Repeat from updated walk location (*$\boldsymbol{n}$ *is undefined if* $p \notin \partial\Omega_{\mathrm{N}}$*)*

term $\mu_{\boldsymbol{v}}$. Importantly, the direction of each estimate evolves over the course of the walk, as it is always aligned with the local direction of $\boldsymbol{\rho}_{\boldsymbol{v}}$. We provide pseudocode for the full directional derivative estimator in Algorithm 1, which retains the same structure as the WoSt algorithm for the solution [Miller et al. 2024b, Algorithm 1].

4.2 Terminating Walks on the Neumann Boundary

As described in Section 3.2.3, the original WoSt algorithm is ill-suited for solving pure Neumann problems—whether for the solution or its derivatives—because random walks continue indefinitely unless artificially truncated. However, unlike the solution u, which is defined only up to an additive constant, the directional derivative $\partial_{\boldsymbol{v}} u$ is uniquely determined. This distinction removes the ambiguity inherent to the solution and suggests that an estimator for $\partial_{\boldsymbol{v}} u$ can, in principle, be constructed without arbitrary termination. The vector-valued function $\boldsymbol{\rho}_{\boldsymbol{v}}$ in our derivative estimator (Equation 13) offers such a mechanism, similar to how the scalar function ρ in Equation 5 facilitates walk termination for Robin boundary conditions.

In more detail, when the radius R of a star-shaped region $\mathrm{St}(x, R)$ is chosen as the distance from x to the closest silhouette point on the reflecting boundary, St may include boundary points for which the reflectance functions become unbounded (Figure 3). For instance, in both 2D and 3D, $\boldsymbol{\rho}_{\boldsymbol{v}}$ has the explicit form

$$\boldsymbol{\rho}_{\boldsymbol{v}} = \boldsymbol{v} - \boldsymbol{v} \cdot \boldsymbol{n} \frac{z - x}{(z - x) \cdot \boldsymbol{n}}, \tag{14}$$

where $\boldsymbol{n}$ denotes the outward normal at the boundary point $z \in \partial\mathrm{St}_{\mathrm{N}}$. The magnitude of this vector diverges as the denominator $(z - x) \cdot \boldsymbol{n}$ approaches zero, *i.e.*, when the view direction from x is nearly perpendicular to $\boldsymbol{n}$. However, by selecting a smaller radius while still ensuring that St remains star-shaped (*i.e.*, every boundary point in St remains visible from its center x), we can restrict the magnitude of $\boldsymbol{\rho}_{\boldsymbol{v}}$ to lie within a desired range, such as $[0, 1]$. This mirrors the strategy used by Miller et al. [2024b] for Robin problems, where the scalar reflectance function ρ is likewise bounded and interpreted as a reflection probability on $\partial\mathrm{St}_{\mathrm{N}}$—hence the term "reflectance."

For derivative estimation, we select the smallest radius such that the corresponding star-shaped region will exclude any boundary point where the reflectance exceeds a prescribed threshold $\rho_{\max}$:

$$R = \min\{\|x - z\| : z \in \partial\Omega_N, |\boldsymbol{\rho}_{\boldsymbol{v}}(x, z)| \geq \rho_{\max}\} \tag{15}$$

We assume $\rho_{\max} = 1$ in Algorithm 1, and provide implementation details for computing R when the boundary $\partial\Omega_N$ is represented by a triangle mesh in Section 5.

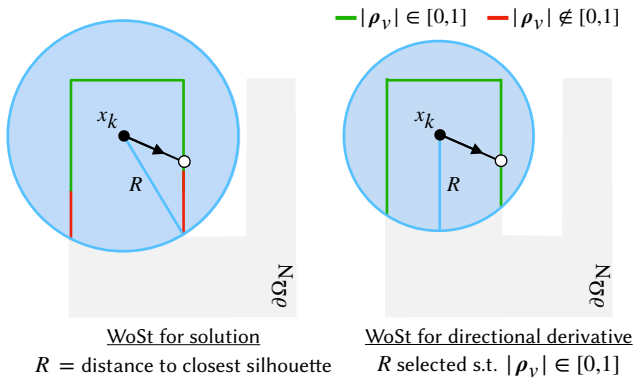


Fig. 3. Unlike the WoSt solution estimator, which sets the star-shaped region radius R to the distance from the walk location x_k to the closest silhouette point on $\partial\Omega_{\mathrm{N}}$ (*left*), the derivative estimator chooses R so that the reflectance magnitude $|\boldsymbol{\rho}_{\boldsymbol{v}}|$ remains within a prescribed range (*right*).

Termination Using Russian Roulette. When $|\boldsymbol{\rho}_{\boldsymbol{v}}| < 1$, we terminate walks with probability $1 - |\boldsymbol{\rho}_{\boldsymbol{v}}|$ at each step (*line 10*, Algorithm 1). If a walk survives, we normalize the reflectance to unit length and continue estimating the directional derivative along $\boldsymbol{\rho}_{\boldsymbol{v}}/|\boldsymbol{\rho}_{\boldsymbol{v}}|$, without scaling the estimate by $|\boldsymbol{\rho}_{\boldsymbol{v}}|$ (*line 11*). This Russian roulette scheme guarantees finite walk lengths even in pure Neumann problems, while avoiding both the bias of arbitrary truncation and the variance from singular kernels.

4.3 Extension to Mixed Dirichlet-Neumann Conditions

While our focus has been on Neumann boundary conditions which are more challenging for Monte Carlo PDE solvers, our derivative estimator also extends naturally to domains with Dirichlet conditions.

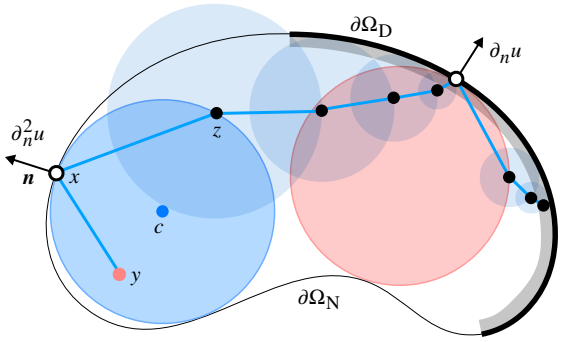


Fig. 4. Following Yu et al. [2024], our method estimates the normal derivative $\partial_n u$ on the Dirichlet boundary $\partial\Omega_D$ by launching a secondary walk, shown in red, from an off-centered sphere tangent to the boundary (Section 4.3). We extend this approach in Section 4.4 to also estimate the second normal derivative $\partial_n^2 u$ on the Neumann boundary $\partial\Omega_N$.

One possible approach is to formulate a boundary integral equation analogous to Equation 9 for the directional derivative, where the star-shaped region St may include portions of $\partial\Omega_D$ as well. We derive such an expression in Appendix B.

In practice, we adopt a simpler strategy more closely aligned with the WoSt solution estimator: we restrict the radius of St so it does not exceed the distance to $\partial\Omega_D$. Any walk that comes within an ε distance of the Dirichlet boundary is projected onto $\partial\Omega_D$, where the tangential component of $\partial_v u$ is set to $\partial_{v_\Gamma} g$, the directional derivative of the Dirichlet data along the tangent vector v_Γ. Since the normal derivative $\partial_n u$ is not prescribed on $\partial\Omega_D$, we follow the strategy of Yu et al. [2024, Section 5.2] and estimate it by launching a secondary WoSt process from an off-centered sphere tangent to the boundary (Figure 4). This construction enables the use of control variates to mitigate kernel singularities on $\partial\Omega_D$, providing lower-variance reconstruction of $\partial_v u$ from its tangential and normal components.

4.4 Higher Order Derivative Estimation

Certain applications, such as the gradient-based optimization of Neumann boundaries (Section 3.4), require estimating higher-order derivatives. In particular, computing the parameter derivative $\dot{u}$ in Equation 7 involves evaluating the second normal derivative $\partial_n^2 u$ and tangential derivative $\nabla u \cdot \nabla_\Gamma V_n$ on the Neumann boundary. While the latter can be estimated using our directional derivative estimator (Equation 13), the former presents an additional challenge.

In this section, we focus specifically on estimating $\partial_n^2 u$ on $\partial\Omega_N$. We do not address other second-order derivatives, such as the full Hessian of u, which are beyond the scope of this work. The basic insight is that since the directional derivative $\partial_n u$ satisfies a Poisson equation, its own normal derivative admits a boundary integral representation similar to Equation 6. In particular, we have

$$\begin{aligned}\partial_n^2 u(x) = &\int_{\partial \mathrm{B}(c,R)} \partial_n P^{\mathrm{B}}(x,z)\ \partial_n u(z)\ \mathrm{d}z \\ &+ \int_{\mathrm{B}(c,R)} \partial_n G^{\mathrm{B}}(x,y)\ \partial_n f(y)\ \mathrm{d}y,\end{aligned} \tag{16}$$

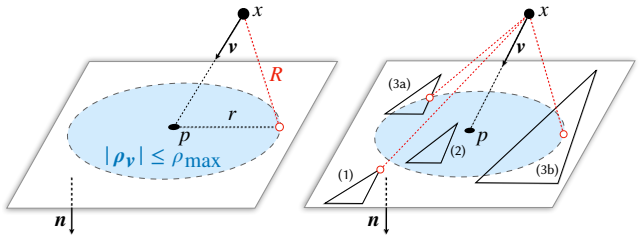


Fig. 5. Geometric construction for determining the radius of a star-shaped region for a plane and triangle. *Left:* The reflectance vector $\boldsymbol{\rho}_v$ lies in the plane and defines a disk of radius r centered at the intersection point p, such that $|\boldsymbol{\rho}_v| \le \rho_{\max}$ inside the disk. The star-shaped region radius R is the distance from x to the disk boundary. *Right:* Three ways a triangle may intersect the disk: (1) lies fully outside, (2) lies fully inside, and (3a,b) partially overlaps. In cases (1) and (3), the triangle constrains the star-shaped radius.

where the ball $\mathrm{B}(c,R)$ is centered at $c \in \Omega$ and is fully contained within the domain. As illustrated in Figure 4, the ball is constructed to be tangent to the Neumann boundary at the evaluation point x, which lies on both $\partial\Omega_N$ and $\partial\mathrm{B}(c,R)$, with outward normal $\boldsymbol{n}$. This formulation allows us to estimate $\partial_n^2 u$ by recursively applying our directional derivative estimator to $\partial_n u(z)$ along the integration boundary. Since the singular kernels $\partial_n P^{\mathrm{B}}(x,z)$ and $\partial_n G^{\mathrm{B}}(x,y)$ in Equation 16 can lead to high variance near the evaluation point x, we also provide a control variate strategy in Appendix D.3 for stable and accurate estimation.

5 IMPLEMENTATION ON TRIANGLE MESHES

As discussed in Section 4.2, selecting an appropriate radius for star-shaped regions is essential for bounding the magnitude of the reflectance at each step of a walk. However, closed-form expressions for this radius are generally unavailable in domains with arbitrary boundary geometry. In this section, we describe how to compute star-shaped region radii on triangle meshes. We begin in Section 5.1 with the case of a single triangle, and generalize to full meshes in Section 5.2, where we accelerate radius queries using a spatial hierarchy. Then, in Section 5.3, we present a specialized edge sampling strategy for computing the second integral in Equation 9, which involves evaluating the directional derivative of known Neumann data and local mean curvature.

5.1 Star-Shaped Region Selection For A Triangle

To compute the radius R of a star-shaped region St centered on x when the Neumann boundary is defined by a single triangle, we first consider the simpler case of an infinite plane. The reflectance vector $\boldsymbol{\rho}_v$ from Equation 14 lies entirely within the plane, as its dot product with the plane's normal $\boldsymbol{n}$ yields zero. Geometrically, the magnitude $|\boldsymbol{\rho}_v|$ defines the radius r of a disk in the plane centered at the point p (Figure 5, *left*), where a ray from x in direction $\boldsymbol{v}$ intersects the plane. A short derivation shows that r satisfies

$$r \le \rho_{\max} \cdot \frac{(p-x)\cdot \boldsymbol{n}}{\boldsymbol{v}\cdot\boldsymbol{n}}, \tag{17}$$

implying that any point z within this disk yields a reflectance magnitude below the threshold $\rho_{\max}$. Outside the disk, reflectance values

exceed the threshold. Hence, for a plane, the radius R of St is given by the minimum distance from x to the disk boundary.

For a triangle that spans only a subset of the plane, we restrict attention to the portion of the disk that overlaps the triangle (Figure 5, *right*). This intersection yields three distinct cases:

(1) **No overlap:** All points in the triangle produce reflectance values above $\rho_{\max}$. To exclude these points from St, we set R to the minimum distance from x to the triangle.

(2) **Full containment:** All points in the triangle satisfy the reflectance constraint. Since the triangle does not place a restriction on St, we set $R = \infty$.

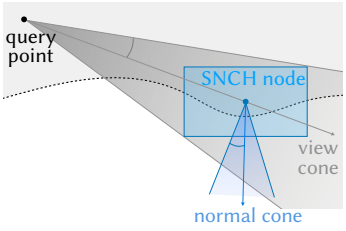
(3) **Partial intersection:** Only a portion of the triangle satisfies the reflectance constraint. In this case, we compute R as the minimum distance from x to the boundary of the intersection between the triangle and the disk:

$$R = \min\{\|x - z\| : z \notin \triangle \cap \bigcirc\}. \tag{18}$$

Practically, this involves evaluating the minimum distance to the disk boundary, as well as to the intersection points between the triangle's edges and the disk boundary.

5.2 Star-Shaped Region Selection For Triangle Meshes

Computing the radius R by evaluating every triangle individually and selecting the minimum is computationally expensive for large meshes. To accelerate both star-shaped region selection and ray intersection against the Neumann boundary (Algorithm 1, *lines 1 & 5*), we adopt a *spatialized normal cone hierarchy* (SNCH) [Johnson and Cohen 2001], which remains unchanged from prior WoSt estimators. This structure augments a standard bounding volume hierarchy with angular bounds, enabling efficient pruning based on both spatial proximity and surface orientation. Specifically, each node in the SNCH stores an axis-aligned bounding box (AABB) along with a *normal cone*, *i.e.*, a cone that bounds all surface normals of the triangles in the node (inset). The cone's axis is the average normal direction, and its half-angle captures the maximum deviation from this axis.



Traversal and Culling. To compute R at a given walk location x, we traverse the SNCH in depth-first order, progressively refining a conservative estimate of the radius. At each node, we construct a *view cone* rooted at x, whose axis points to the node's centroid and whose half-angle tightly encloses the node's AABB. Following the closest silhouette point query procedure of Sawhney et al. [2023, Section 5.1.2], we first check whether the view cone and normal cone admit a pair of mutually orthogonal directions [Sawhney et al. 2023, Algorithm 4]. If so, the node may contain silhouette edges and must be visited. If not, we compute a conservative upper bound for the reflectance magnitude $|\boldsymbol{\rho}_v|$ using the node's spatial and angular bounds, as described below. If this bound is smaller than the threshold $\rho_{\max}$ or if the node lies entirely outside the current best radius estimate, it is safely culled. For the surviving leaf nodes, we apply the triangle-level procedure from Section 5.1, potentially tightening the global minimum. This traversal strategy substantially reduces the number of reflectance evaluations and offers significant speedups over brute-force linear traversal.

Upper Bound For Reflectance Magnitude. To conservatively bound reflectance over an entire SNCH node, we analyze the squared magnitude of Equation 14:

$$\begin{aligned} &\left|\boldsymbol{\rho}_v(x,z)\right|^2 = 1 - 2ab + a^2, \\ &\text{where} \quad a = \frac{\boldsymbol{v}\cdot\boldsymbol{n}}{\boldsymbol{d}\cdot\boldsymbol{n}}, \quad b = \boldsymbol{v}\cdot\boldsymbol{d}, \quad \boldsymbol{d} = \frac{z-x}{\|z-x\|}. \end{aligned} \tag{19}$$

The dot products in a and b vary over angular intervals determined by the node's normal cone and the view cone. Using interval arithmetic, the intervals $\boldsymbol{d}\cdot\boldsymbol{n} \in [\cos_{\min}\alpha, \cos_{\max}\alpha]$, $\boldsymbol{v}\cdot\boldsymbol{n} \in [\cos_{\min}\beta, \cos_{\max}\beta]$, and $\boldsymbol{v}\cdot\boldsymbol{d} \in [\cos_{\min}\gamma, \cos_{\max}\gamma]$ yield conservative bounds for a and b:

$$a \in \left[\frac{\cos_{\min}\beta}{\cos_{\max}\alpha}, \frac{\cos_{\max}\beta}{\cos_{\min}\alpha}\right], \quad b \in [\cos_{\min}\gamma, \cos_{\max}\gamma]. \tag{20}$$

Applying these expressions to Equation 19 gives an upper bound on the squared reflectance:

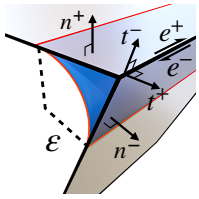
$$\begin{aligned} \left|\boldsymbol{\rho}_v\right|^2_{\text{upper}} = \; & 1 - 2\min\left(a_{\min}b_{\min},\, a_{\min}b_{\max},\, a_{\max}b_{\min},\, a_{\max}b_{\max}\right) \\ & + \max\left(a^2_{\min},\, a^2_{\max}\right). \end{aligned} \tag{21}$$

If $\left|\boldsymbol{\rho}_v\right|^2_{\text{upper}} < \rho^2_{\max}$, the node cannot influence the radius estimate and is culled; otherwise, it must be visited.

5.3 Resolving Derivative Discontinuities

The function η_v (Equation 12) on the Neumann boundary involves the surface derivative $\partial_{v_\Gamma} h$ and the mean curvature H. Both become singular along mesh edges where the surface normal changes abruptly, yielding Dirac delta contributions supported only on edges. (Note that H vanishes within each flat triangle.) Because these contributions lie on sets of measure zero, standard Monte Carlo sampling over triangles can fail to capture them, leading to biased directional derivative estimates [Li et al. 2018].

Line Integral Formulation. To correct this bias, we augment the surface integral $\int_{\partial \mathrm{St}_\mathrm{N}} G^\mathrm{B}\eta_v$ in Equation 9 with an additional line integral over the set of Neumann boundary edges $\partial\mathrm{St}^\mathrm{E}_\mathrm{N}$ inside a star-shaped region St. This is done by mollifying the geometry: each sharp edge is replaced by a narrow band of width ε, across which surface normals transition smoothly and the function η_v becomes regular (inset). In Appendix C, we show the contribution of this band converges, in the limit $\varepsilon \to 0$, to the following line integral:



$$\int_{\partial\mathrm{St}^\mathrm{E}_\mathrm{N}} G^\mathrm{B}(x,z)\;\left[\boldsymbol{v}\cdot\boldsymbol{t}^+(z)h^+(z) + \boldsymbol{v}\cdot\boldsymbol{t}^-(z)h^-(z)\right]\mathrm{d}l, \tag{22}$$

where z lies on an edge shared by triangles with normals $\boldsymbol{n}^+$ and $\boldsymbol{n}^-$. Letting $\boldsymbol{e}^+$ and $\boldsymbol{e}^-$ be unit tangent vectors along opposite edge

directions, the corresponding in-plane edge normals are defined as

$$t^+ = \frac{e^+ \times n^+}{\|e^+ \times n^+\|}, \quad t^- = \frac{e^- \times n^-}{\|e^- \times n^-\|}. \tag{23}$$

The values $h^+(z)$ and $h^-(z)$ denote the Neumann data from the two adjacent triangles. Although z lies on a shared edge, this formulation correctly accounts for discontinuities in both the surface geometry and the boundary data.

Edge Sampling. Fortunately, sampling Equation 22 requires no additional machinery beyond an SNCH-based traversal used in prior WoSt estimators. We reuse the point sampling query of Sawhney et al. [2023, Section 5.2] to efficiently identify Neumann boundary edges within St, then uniformly sample a point z along one such edge. In Appendix D.2, we also describe a control variate strategy to mitigate singular behavior in the Green's function $G^{\mathrm{B}}(x, z)$ as a sampled point z along an edge approaches the query location x.

Additional considerations. While sampling the above line integral resolves singularities from η_v, other sources of irregularity remain. In particular, the directional derivative $\partial_\rho u$ may also exhibit limited smoothness near sharp concave corners on $\partial\Omega_{\mathrm{N}}$. We revisit this issue in Section 6.5.

6 RESULTS

We implement our solver using Dr.Jit [Jakob et al. 2022] for parallel execution of random walks on the GPU. For star-shaped region selection, ray intersections, and distance and point sampling queries, we use the spatialized normal cone hierarchy from the fcpw library [Sawhney 2021]. The hierarchy is constructed on the CPU, while traversal for all queries is performed on the GPU. All experiments were run on a workstation with an NVIDIA RTX 4090 GPU.

We begin by validating our method on synthetic test problems (Section 6.1). We then demonstrate new capabilities enabled by our approach, including reconstructing the solution to pure Neumann problems (Section 6.2), simulating divergence-free magnetic fields (Section 6.3), and optimizing Neumann boundaries via parameter derivative estimation (Section 6.4). Finally, Section 6.5 discusses the unique challenges of computing derivatives in polyhedral domains.

6.1 Validation

We validate our method on two synthetic test cases, shown in Figures 6 and 7, using domains defined by 2D polylines (first row) and 3D triangle meshes (second row), respectively. For the 3D model, we restrict evaluation to a 2D slice through the volume. In each case, we prescribe an analytical function $\bar{u}$ within the domain and compute the corresponding source term and boundary conditions to define a Poisson problem:

$$\begin{aligned} \Delta u &= \Delta \bar{u} && \text{on } \Omega, \\ u &= \bar{u} && \text{on } \partial\Omega_{\mathrm{D}}, \\ \partial_n u &= \partial_n \bar{u} && \text{on } \partial\Omega_{\mathrm{N}}. \end{aligned} \tag{24}$$

We evaluate the accuracy of our method by comparing its estimates of the directional derivative $\partial_x u$ to the exact value $\partial_x \bar{u}$. We also compare our results to those produced by the baseline gradient estimator described in Section 3.3.

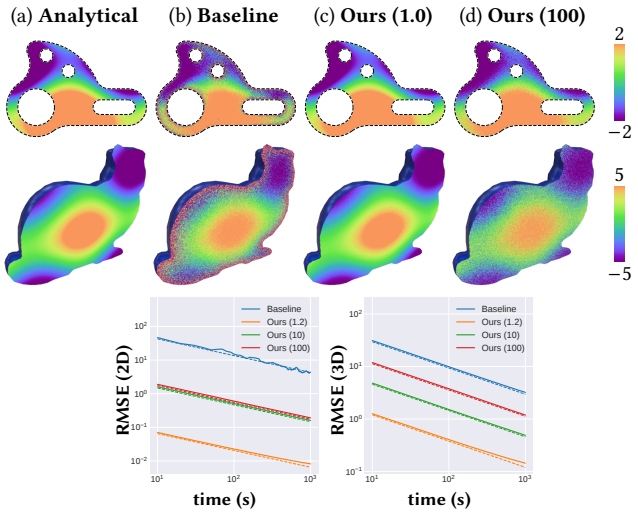


Fig. 6. Estimation of directional derivatives for an analytically defined pure Neumann problem. The baseline WoSt estimator (Section 3.3) in column (b) exhibits high variance and bias, especially near the boundary, with walks truncated after a fixed number of steps (here, 128). In contrast, using our method with the reflectance threshold $\rho_{\max} = 1$ in column (c) yields more accurate results for the same compute budget. Larger thresholds values, such as $\rho_{\max} = 100$ in column (d), lead to higher noise.

6.1.1 *Derivative Estimation For Pure Neumann Problems.*

As discussed in Section 3.2.3, the standard WoSt estimator faces fundamental challenges in pure Neumann settings. Walks lack natural termination criteria, and practical fixes such as truncating walks after a fixed number of steps or applying Tikhonov regularization introduce bias. In addition, the baseline gradient estimator differentiates a kernel that is singular on the boundary (Equation 6), leading to elevated variance and instability in nearby regions (Figure 6(b)).

Our method avoids these pitfalls by reformulating the directional derivative as a boundary integral over a star-shaped region weighted by a reflectance function (Equation 9). This eliminates the need to evaluate singular kernels and enables unbiased walk termination via Russian roulette (Section 4.2). As a result, our estimator achieves lower variance than the baseline throughout the domain, including near the boundary, and delivers higher accuracy at comparable cost. Figure 6(d) shows that setting a reflectance threshold $\rho_{\max} > 1$, corresponding to a larger star-shaped radius, produces higher noise.

6.1.2 *Derivative Estimation for Mixed Dirichlet–Neumann Conditions.*

For problems with mixed boundary conditions, we follow the approach described in Section 4.3: when a random walk enters the ε-shell around a Dirichlet boundary, we launch a secondary walk to estimate the normal derivative on $\partial\Omega_{\mathrm{D}}$, following Yu et al. [2024].

Although the baseline estimator can terminate walks upon reaching $\partial\Omega_{\mathrm{D}}$, it still exhibits high variance near both boundary types. This effect is especially pronounced near Neumann boundaries, where walks tend to be significantly longer. The resulting disparity in walk lengths leads to uneven variance across the domain, as shown in Figure 7. Our method outperforms the baseline, achieving lower variance in equal-time.

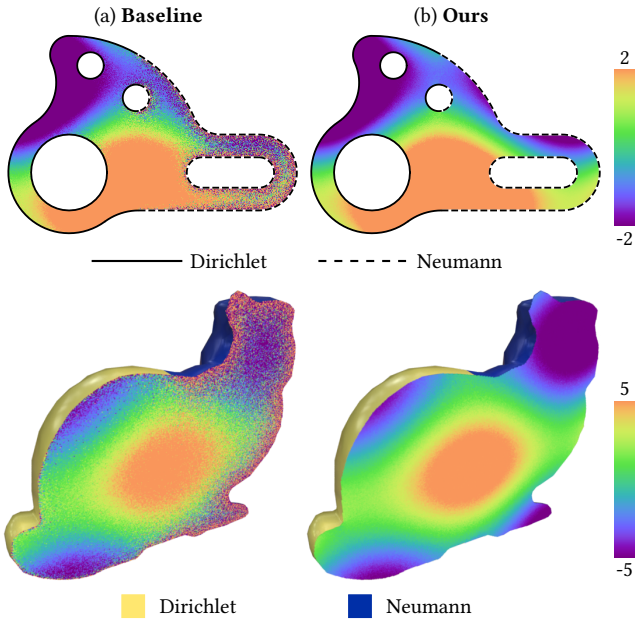


Fig. 7. Directional derivative estimation under mixed boundary conditions. Walks terminate upon reaching the Dirichlet boundary, resulting in shorter trajectories than in the pure Neumann case. Nonetheless, the baseline estimator remains inefficient near the Neumann boundary, where our method yields lower variance for the same compute budget.

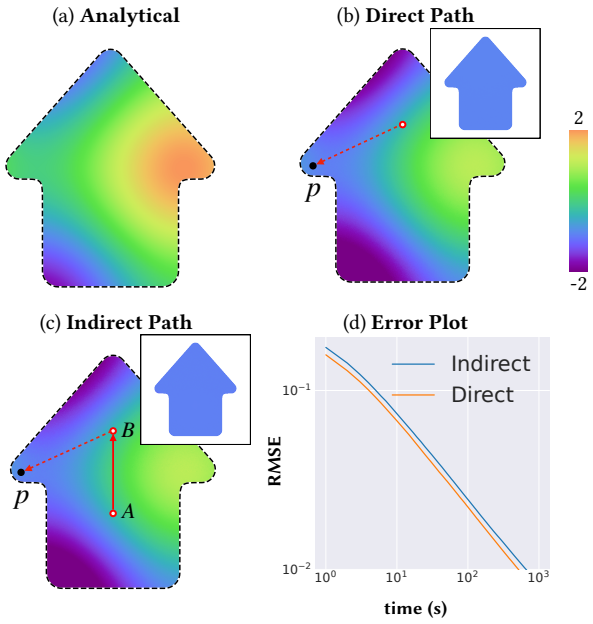


Fig. 8. *Top row:* To reconstruct the solution to a pure Neumann problem, we first fix its value at a single point in the domain. The estimated results then match the analytical reference pointwise, up to an additive constant anywhere in the domain (inset). *Bottom row:* In regions not directly reachable from the pinned point A, we can recover the solution by chaining multiple line integrals, *e.g.*, from A to B, then from B to all other points.

6.1.3 Potential Flow Simulation. As a more challenging test case, Figure 1 shows a potential flow simulation inside a bounding box with marine life spanning a wide range of spatial scales. The potential field u is governed by the Laplace equation with prescribed Neumann boundary conditions: an inflow condition $\partial_n u = -1$ is applied to the upstream face of the box, an outflow condition $\partial_n u = 1$ to the downstream face, and zero Neumann conditions $\partial_n u = 0$ to the remaining faces and the marine life, modeled as a triangle mesh with 337,744 primitives. SNCH construction on the CPU required 1.4 seconds. We traced 100 streamlines, each with 100 points and 1,000 walks per point, for a total runtime of just under two hours.

Unlike traditional PDE solvers requiring volumetric discretizations adapted to the finest geometric features, our method can estimate the gradient field at arbitrary resolutions, enabling localized streamline tracing across multiple scales. Each component–$\partial_x u$, $\partial_y u$, and $\partial_z u$–is estimated independently with our derivative estimator, while the baseline computes all three simultaneously in a single walk. Even so, our approach yields more robust derivatives and higher-quality streamlines under equal-time constraints. As shown in Figure 1, the streamlines curve smoothly around the marine life, as expected in potential flow.

6.2 Reconstructing Solutions to Pure Neumann Problems

A defining feature of pure Neumann problems is that their solutions are unique only up to an additive constant. This non-uniqueness poses a fundamental challenge for the original WoSt method [Sawhney et al. 2023], which aims to estimate the solution directly. We take a different approach: since derivatives remain uniquely defined, we reconstruct the solution by integrating its gradient along paths originating from an arbitrary reference point. Practical advantages of our derivative estimator, such as unbiased early termination of walks, carry over naturally to this reconstruction scheme.

To reconstruct the solution at a query point $p \in \Omega$, we first select a reference point p_0 in the domain and fix the solution value there, typically setting $u(p_0) = 0$ for simplicity. The solution at p is then recovered by integrating the gradient along a path $\mathbf{x}$ from p_0 to p:

$$u(p) = u(p_0) + \int_{p_0}^{p} \nabla u(\mathbf{x}) \cdot \mathrm{d}\mathbf{x}. \tag{25}$$

In Figure 8(b), we encode the integration path with linear segments and uniformly sample it to evaluate directional derivatives along the direction from p_0 to p. This reconstruction accurately recovers the solution across the domain, matching the ground-truth analytical function up to an additive constant (shown in the inset).

More generally, the integration path between the reference and query points can be arbitrary; any parameterized curve suffices. To validate this path independence, Figure 8(c) shows solutions reconstructed with an indirect path consisting of two linear segments, for simplicity. For regions not directly visible from the pinned point A, the path is routed through an intermediate point B. The reconstructed values again match the reference solution up to an additive constant, though longer paths incur slightly higher variance. Designing a principled scheme to automatically connect query points

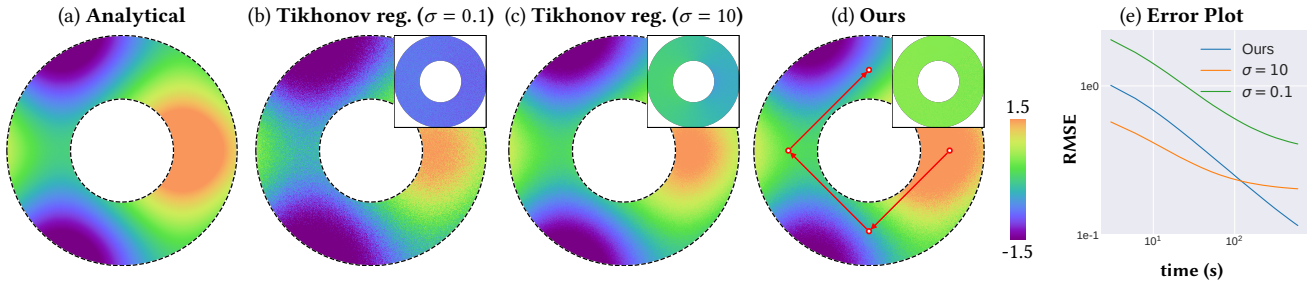


Fig. 9. Our reconstruction of the pure Neumann problem (d) exhibits lower noise than a Tikhonov regularization with a small absorption coefficient (b), and reduced bias compared to a large absorption coefficient (c). In (d), the solution is computed by integrating estimated derivatives along linear segments from each query point to its nearest visible reference point (red dot). The insets show pointwise errors relative to the analytical solution.

or regions of interest to the pinned point via parameterized curves remains an open direction for future work.

Figure 9 highlights the practical advantages of our method over the original WoSt estimator, which relies on a Tikhonov regularization to handle pure Neumann problems. As discussed in Section 3.2.3, small absorption coefficients preserve accuracy but amplify noise, while larger coefficients reduce variance at the cost of significant bias. In contrast, our approach avoids this tradeoff entirely, achieving both low noise and low bias in the reconstructed solution.

6.3 Computing Divergence-Free Vector Fields

Helmholtz decomposition is a standard technique for computing divergence-free vector fields in physical simulations [Bhatia et al. 2013; Nabizadeh et al. 2021]. It expresses an arbitrary vector field $\mathbf{b}$ as the sum of a divergence-free component $\mathbf{F}$ and the gradient of a scalar potential u, *i.e.*, $\mathbf{b} = \mathbf{F} + \nabla u$. The divergence-free component is then recovered as $\mathbf{F} = \mathbf{b} - \nabla u$, where u satisfies the Poisson equation $\Delta u = \nabla \cdot \mathbf{b}$. We perform this decomposition for pure Neumann problems by directly estimating the gradient ∇u, thereby recovering $\mathbf{F}$ without explicitly solving for u.

Magnetostatics. A practical instance of a Helmholtz decomposition arises in the design of industrial magnets [Wolfram Research 2025]. In this setting, the magnetic flux density is given by $\mathbf{B} = \mu_0(\mathbf{H} + \mathbf{M})$, where μ_0 is the vacuum permeability, $\mathbf{M}$ is the magnetization of a permanent magnet, and $\mathbf{H}$ is the demagnetizing field. The field $\mathbf{H}$ is derived from a magnetic scalar potential ϕ_m via $\mathbf{H} = -\nabla\phi_m$, where ϕ_m satisfies the Poisson equation $\Delta\phi_m = -\nabla \cdot \mathbf{M}$.

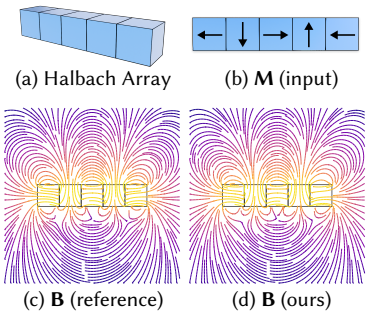
In practice, the magnetization $\mathbf{M}$ is often discontinuous across the surface of a magnet, denoted $\partial\mathcal{M}$. This discontinuity leads to a decomposition of the source term $-\nabla \cdot \mathbf{M}$, representing the effective magnetic charge density, into two physically distinct components: a volume charge density $\rho_m = -\nabla \cdot \mathbf{M}$ inside the magnet, and a surface charge density $\sigma_m = \mathbf{M} \cdot \boldsymbol{n}$ concentrated at the interfaces. The source term in the integral equation for ϕ_m (Equation 3) therefore sums contributions from these distinct charge distributions:

$$\int_{\mathrm{St}} G^{\mathrm{B}}(x,y)\rho_m(y)\mathrm{d}y + \int_{\mathrm{St}\cap\partial\mathcal{M}} G^{\mathrm{B}}(x,y)\sigma_m(y)\mathrm{d}y. \tag{26}$$

Similarly, the source integral for the directional derivative $\partial_v\phi_m$ in Equation 9 becomes:

$$\int_{\mathrm{St}} G^{\mathrm{B}}(x,y)\partial_v\rho_m(y)\mathrm{d}y + \int_{\mathrm{St}\cap\partial\mathcal{M}} \partial_v G^{\mathrm{B}}(x,y)\sigma_m(y)\mathrm{d}y. \tag{27}$$

Field Estimation. We estimate the integrals above using the same sampling strategies as those employed for the source and Neumann integrals, respectively, in Sawhney et al. [2023, Sections 4.5 & 4.6]. We validate our solver on a *Halbach array*, a special configuration of permanent magnets that enhances the magnetic field on one side while nearly canceling it on the other. In the inset, the array is enclosed within a large bounding sphere with zero Neumann boundary conditions. Using the specified magnetization pattern (b) of the array as input, we compute the resulting magnetic field (d), which closely matches the reference field lines (c) calculated using Magpylib [Ortner and Coliado Bandeira 2020].



In Figure 10, we compute the magnetic flux density in an industrial magnetic bearing [COMSOL 2025] with extremely fine-scale surface imperfections. The magnet is represented as a triangle mesh with 1,189,664 primitives. Building a bounding volume hierarchy for point sampling of the source integral (Equation 27) required 2.5 seconds, while SNCH construction for the bounding sphere around the magnet took only a few milliseconds. We then sampled approximately 27,000 points along the streamlines, performing 10,000 walks per point, for a total runtime of about two minutes.

Finite element methods face significant challenges in this setting, as standard tetrahedral meshing tools like TetGen [Si 2015] cannot robustly process non-manifold or self-intersecting surface meshes. While mesh repair tools such as MeshFix [Attene 2010] can correct connectivity issues, they often distort the geometry of critical features–such as cracks–in the process. Even after repair, TetGen fails to generate a valid volumetric mesh for this magnetic bearing. More robust meshing algorithms like fTetWild [Hu et al. 2020] handle degenerate inputs more effectively but still fail to capture

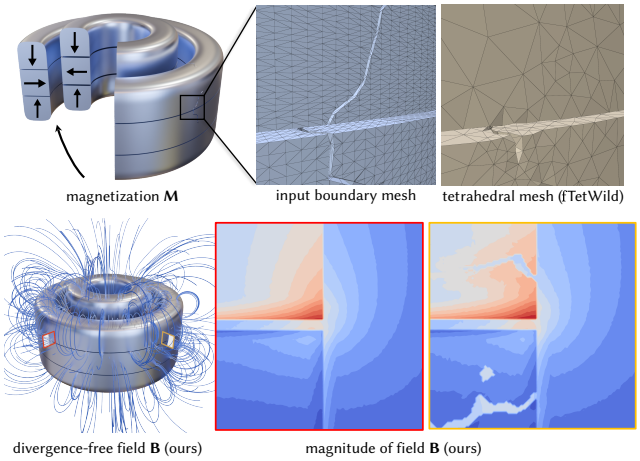


Fig. 10. Magnetic flux density in industrial magnetic bearings. Our Monte Carlo method accurately computes the divergence-free field **B** in the far field while also resolving fine-scale variations near surface defects such as cracks (*bottom row*). In contrast, finite element meshing (here using fTetWild with default settings) may fail to capture such geometric detail (*top row, right*), leading to inaccurate simulation results in critical regions.

the intricate crack geometry with default settings (Figure 10, top right). Although fTetWild's epsilon parameter can be reduced to resolve smaller features, it must be set at or below the smallest geometric scale in the domain, potentially leading to excessive memory consumption and high computational cost.

Boundary element methods, which avoid volumetric meshing altogether, have seen substantial performance improvements in recent years [Chen et al. 2024, 2025], but they do not apply directly to the source-dominated problem considered here. In contrast, our Monte Carlo method sidesteps meshing challenges entirely and resolves the magnetic field both near the crack and in the far field, capturing fine-scale geometric detail with minimal preprocessing.

6.4 Gradient-Based Optimization of Neumann Boundaries

PDE-constrained shape optimization is central to inverse design and simulation. While Monte Carlo solvers have only recently begun to emerge for such problems, differentiable optimization with Neumann boundaries has remained largely unexplored due to the absence of reliable estimators for first- and second-order spatial derivatives. This gap is particularly relevant in applications such as magnetic field shaping, thermal insulation design, and structural load redistribution, where Neumann conditions naturally model fluxes, stresses, and applied forces. Our method introduces the first tractable Monte Carlo strategy for estimating parameter derivatives of a Poisson equation in this setting (Equation 7).

Problem setup. As a simple illustrative example, Figure 12 depicts the optimization of a fish-shaped Neumann boundary $\partial\Omega_{\mathrm{N}}$, parameterized by its position $\mathbf{t}$ and scale r, *i.e.*, $\pi = (\mathbf{t}, r)$. The objective is to match the solution $u(x, \pi)$ of a Poisson equation to prescribed target values at a sparse set of measurement points (red dots), mimicking the task of tuning an insulating region embedded in a conductive

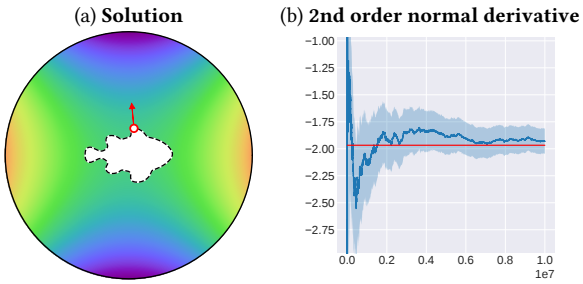


Fig. 11. We evaluate our estimator for the second-order normal derivative $\partial_n^2 u$ on a PDE with a known analytical solution. The plot shows the estimated mean and confidence interval for a fixed evaluation point as the number of samples increases.

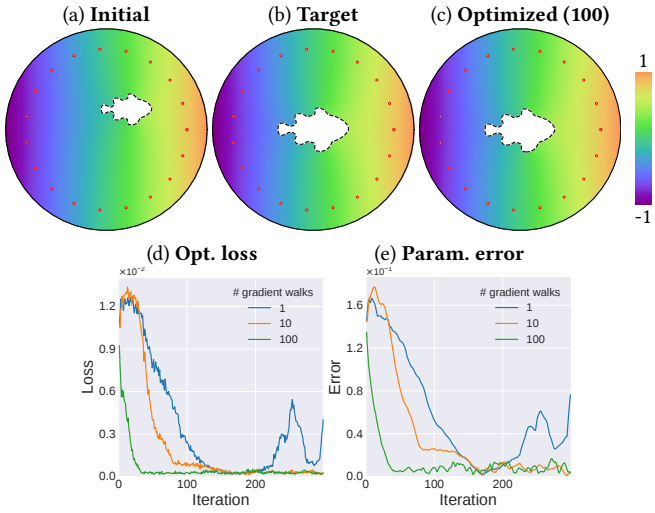


Fig. 12. *Top row:* We optimize the position and scale of a fish-shaped Neumann boundary by minimizing an L^2 loss between the solution u to a Poisson equation and target values specified at a sparse set of points in the domain (red dots). The optimized result, obtained using 100 walks per iteration to estimate the spatial derivatives of u, is shown on the top right (c). *Bottom row:* Ablation study on the number of walks used per iteration to estimate spatial derivatives of u. We plot the optimization loss and parameter error for 1, 10, and 100 such walks, while keeping the number of walks for the parameter derivatives $\dot{u}$ fixed.

material. For this problem, we minimize the shape functional used by Miller et al. [2024a, Section 4],

$$\mathcal{L}(\pi) = \int_{\Omega(\pi)} \mathrm{M}(x)\mathrm{L}(u(x,\pi))\,\mathrm{d}x, \tag{28}$$

where L is a differentiable loss function and M is a mask that localizes the objective to regions of interest. In our experiment, we use an L2 loss $\mathrm{L}(u(x,\pi)) = \|u(x,\pi) - u_{\mathrm{target}}(x)\|^2$. The mask M is defined as a collection of Dirac delta functions at discrete measurement points x_i, reducing the integral to the discrete sum

$$\mathcal{S}(\pi) = \sum_i \left\|u(x_i,\pi) - u_{\mathrm{target}}(x_i)\right\|^2. \tag{29}$$

Minimizing $\mathcal{S}(\pi)$ with respect to the parameters π–and thereby optimizing the Neumann boundary they define–requires evaluating the parameter derivatives $\dot{u} := \partial u/\partial \pi$. These derivatives quantify how the solution responds to infinitesimal changes in the boundary geometry, and are essential for parameter updates via stochastic gradient descent. In Figure 12, we fix the outer Dirichlet boundary $\partial\Omega_{\mathrm{D}}$ (the circle) and optimize only the interior fish-shaped boundary with zero Neumann conditions, $\partial_n u = 0$. However, our method naturally extends to problems involving both boundary types.

Estimation Strategy. In principle, the parameter derivatives $\dot{u}$ can be computed by applying the standard WoSt estimator to the differential Poisson equation in Equation 7. However, the Neumann conditions for this PDE involve spatial derivatives of the primal solution u–specifically, ∇u and $\partial_n^2 u$–which must be estimated on the fly. A naive approach would launch nested random walks at every step of a differential walk to compute these quantities, using our directional and second-order derivative estimators from Sections 4.1 and 4.4, respectively. To avoid the resulting combinatorial explosion in sampling cost, we instead propose a "two-pass" algorithm:

(1) **Precomputation Pass:** We first uniformly sample points on the Neumann boundary $\partial\Omega_{\mathrm{N}}$ and estimate ∇u and $\partial_n^2 u$ using our derivative estimators–Figure 11 confirms reliable convergence of the latter. We cache and index these estimates (*e.g.*, with a k-d tree) for fast retrieval.

(2) **Differential Pass:** We then solve for $\dot{u}$ using WoSt, retrieving cached spatial derivatives on $\partial\Omega_{\mathrm{N}}$ that fall within the star-shaped region associated with the current walk location. This eliminates the need for costly nested walks during estimation.

Figure 12 demonstrates successful convergence of our example inverse problem: the optimized Neumann boundary closely matches the target position and radius. To assess the impact of noise in the spatial derivative estimates, we vary the number of walks used per iteration to estimate them (1, 10, 100), while keeping the number of walks for the parameter derivatives $\dot{u}$ fixed. The results show that optimization can diverge when spatial derivative estimates are too noisy (*e.g.*, with only one walk), but typically succeeds when the noise level is more moderate.

More broadly, the convergence of an inverse solver depends on many factors beyond gradient quality, including the choice of loss function, regularization or preconditioning strategies, optimization algorithm, and learning rate schedule. Although our demonstration involves a simple geometry with few parameters, it highlights the feasibility of Monte Carlo–based gradient estimation as a foundation for scalable, gradient-driven design with parameterized Neumann boundary constraints. Extending this approach to complex geometries and higher-dimensional parameter spaces is an important direction for future work.

6.5 Sensitivity of Derivative Estimation to Sharp Corners

Our method builds on the key insight that, just as the solution u to a Poisson equation can be estimated using random walks, so too can its directional derivative $\partial_v u$. This idea has also been explored more broadly in the context of PDE-constrained optimization [Yılmazer et al. 2022; Yu et al. 2024; Miller et al. 2024a], and it forms the foundation of our approach.

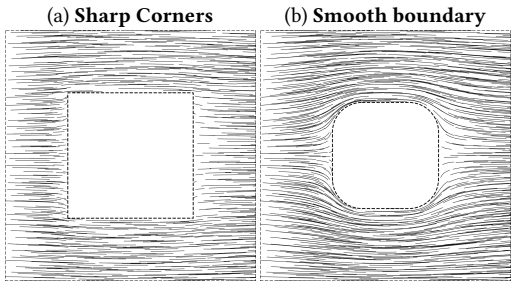


Fig. 13. *Left*: Our method produces inaccurate derivative estimates in the presence of sharp corners. This limitation is evident in the potential flow simulation as streamlines fail to correctly curve around the boundary. *Right*: When the corners are rounded, the estimated gradients yield streamlines that closely match the expected flow.

However, while the standard WoSt estimator exhibits stable convergence when estimating u on polyhedral domains, our method–which directly estimates $\partial_v u$–can struggle in the vicinity of sharp corners or edges. This is because on piecewise-flat domains such as triangle meshes, the solution typically remains continuous up to the boundary, but its directional derivatives can become unbounded–a phenomenon known as the *corner singularity problem* [Grisvard 2011]. This issue is especially pronounced at reentrant (concave) corners, where abrupt changes in boundary orientation induce steep gradients in the solution. Derivative estimates are particularly sensitive in such regions, as differentiation tends to amplify irregularities in the underlying solution.

Moreover, the boundary integral equation introduced in Section 4.1 to estimate $\partial_v u$ does not account for geometric singularities *inside* star-shaped regions, leading to biased derivative estimates near sharp features. Figure 13 illustrates this limitation in a potential flow simulation: sharp corners (left) produce streamlines that exhibit physically implausible behavior, failing to bend naturally around the obstacle with zero Neumann boundary conditions. In contrast, rounding the corners (right) restores smooth flow patterns that align with physical expectations.

The challenges posed by geometric singularities are not unique to our method. In the finite element literature, reentrant corners are known to degrade accuracy, and are typically mitigated through partial remedies such as adaptive mesh refinement or higher-order basis functions. A promising direction for improving our estimator is to mollify the boundary geometry near concave corners, thereby reducing singular behavior and improving derivative accuracy.

7 CONCLUSION

We introduced a Monte Carlo framework based on the walk on stars algorithm for estimating spatial and parameter derivatives of Poisson equations. Our method is particularly effective in Neumann-dominated problems, unlocking new capabilities including principled solution reconstruction, divergence-free field estimation, and shape optimization with parametrically defined boundaries. At the same time, it inherits limitations of the underlying WoSt

estimator–most notably, slower progress near concave boundary regions, where random walks take smaller steps. We expect future improvements to the base algorithm to extend naturally to our derivative estimators. Future directions include accelerating walks near concave boundaries, reducing bias through boundary mollification, and extending the framework to broader classes of elliptic PDEs.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their constructive feedback, and Ken Museth and Aaron Lefohn for their support. This work began while Zihan Yu and Bailey Miller were interns at NVIDIA, and is partially supported by NSF grants 2239627 and 2504890, and a National Institute of Food and Agriculture award 2023-67021-39073.

REFERENCES

Grégoire Allaire, Charles Dapogny, and Pascal Frey. 2014. Shape optimization with a level set based mesh evolution method. *Computer Methods in Applied Mechanics and Engineering* 282 (2014), 22–53.

Marco Attene. 2010. A lightweight approach to repairing digitized polygon meshes. *The Visual Computer* 26 (11 2010), 1393–1406. https://doi.org/10.1007/s00371-010-0416-3

Ghada Bakbouk and Pieter Peers. 2023. Mean Value Caching for Walk on Spheres. In *Eurographics Symposium on Rendering*. The Eurographics Association. https://doi.org/10.2312/sr.20231120

Mégane Bati, Stéphane Blanco, Christophe Coustet, Vincent Eymet, Vincent Forest, Richard Fournier, Jacques Gautrais, Nicolas Mellado, Mathias Paulin, and Benjamin Piaud. 2023. Coupling Conduction, Convection and Radiative Transfer in a Single Path-Space: Application to Infrared Rendering. *ACM Trans. Graph.* 42, 4, Article 79 (jul 2023), 20 pages. https://doi.org/10.1145/3592121

Harsh Bhatia, Gregory Norgard, Valerio Pascucci, and Peer-Timo Bremer. 2013. The Helmholtz-Hodge Decomposition—A Survey. *IEEE Transactions on Visualization and Computer Graphics* 19, 8 (Aug. 2013), 1386–1404. https://doi.org/10.1109/TVCG.2012.316

Jiong Chen, Florian Schaefer, and Mathieu Desbrun. 2024. Lightning-fast Method of Fundamental Solutions. *ACM Trans. Graph.* 43, 4, Article 77 (July 2024), 16 pages. https://doi.org/10.1145/3658199

Jiong Chen, Florian Schäfer, and Mathieu Desbrun. 2025. Lightning-fast Boundary Element Method. *ACM Trans. Graph.* 44, 4, Article 38 (July 2025), 14 pages. https://doi.org/10.1145/3731196

COMSOL. 2025. Axial Magnetic Bearing Using Permanent Magnets (Application ID: 14367). https://www.comsol.com/model/axial-magnetic-bearing-using-permanent-magnets-14367.

Martin Costabel. 1987. Principles of boundary element methods. *Computer Physics Reports* 6, 1-6 (1987), 243–274.

Fernando de Goes and Mathieu Desbrun. 2024. Stochastic Computation of Barycentric Coordinates. *ACM Trans. Graph.* 43, 4, Article 42 (July 2024), 13 pages. https://doi.org/10.1145/3658131

Auguste De Lambilly, Gabriel Benedetti, Nour Rizk, Chen Hanqi, Siyuan Huang, Junnan Qiu, David Louapre, Raphael Granier De Cassagnac, and Damien Rohmer. 2023. Heat Simulation on Meshless Crafted-Made Shapes. In *Proceedings of the 16th ACM SIGGRAPH Conference on Motion, Interaction and Games (MIG '23)*. Association for Computing Machinery, New York, NY, USA, Article 9, 7 pages. https://doi.org/10.1145/3623264.3624457

Pierre Grisvard. 2011. *Elliptic problems in nonsmooth domains.* SIAM.

Antoine Henrot and Michel Pierre. 2018. *Shape Variation and Optimization: A Geometrical Analysis.* EMS Press. 379 pages. https://doi.org/10.4171/178

Raymond M. Hicks and Preston A. Henne. 1977. Wing design by numerical optimization. In *American Institute of Aeronautics and Astronautics, Aircraft Systems and Technology Meeting, Seattle, Wash.; United States; 22-24 Aug. 1977.*

Yixin Hu, Teseo Schneider, Bolun Wang, Denis Zorin, and Daniele Panozzo. 2020. Fast tetrahedral meshing in the wild. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 117–1.

Tianyu Huang, Jingwang Ling, Shuang Zhao, and Feng Xu. 2025. Guiding-Based Importance Sampling for Walk on Stars. *ACM Trans. Graph.* (2025). https://doi.org/10.1145/3721238.3730593 Proc. SIGGRAPH 2025.

Peter Hunter and Andrew Pullan. 2001. Fem/bem notes. *Department of Engineering Science, The University of Auckland, New Zeland* (2001).

Pranav Jain, Ziyan Qu, Peter Yichen Chen, and Oded Stein. 2024. Neural Monte Carlo Fluid Simulation. *ACM Trans. Graph.* 43 (July 2024). https://doi.org/10.1145/3641519.3657438

Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, and Delio Vicini. 2022. Dr.Jit: A Just-In-Time Compiler for Differentiable Rendering. *Transactions on Graphics (Proceedings of SIGGRAPH)* 41, 4 (July 2022). https://doi.org/10.1145/3528223.3530099

David E. Johnson and Elaine Cohen. 2001. Spatialized normal come hierarchies. In *Proceedings of the 2001 Symposium on Interactive 3D Graphics (I3D '01)*. Association for Computing Machinery, New York, NY, USA, 129–134. https://doi.org/10.1145/364338.364380

Tzu-Mao Li, Miika Aittala, Fredo Durand, and Jaakko Lehtinen. 2018. Differentiable Monte Carlo Ray Tracing through Edge Sampling. *ACM Transactions on Graphics (TOG)* (2018).

Zilu Li, Guandao Yang, Xi Deng, Christopher De Sa, Bharath Hariharan, and Steve Marschner. 2023. Neural Caches for Monte Carlo Partial Differential Equation Solvers. In *SIGGRAPH Asia 2023 Conference Papers*. 1–10.

Zilu Li, Guandao Yang, Qingqing Zhao, Xi Deng, Leonidas Guibas, Bharath Hariharan, and Gordon Wetzstein. 2024. Neural Control Variates with Automatic Integration. In *ACM SIGGRAPH 2024 Conference Papers* (Denver, CO, USA) *(SIGGRAPH '24)*. Association for Computing Machinery, New York, NY, USA, Article 10, 9 pages. https://doi.org/10.1145/3641519.3657395

Bailey Miller, Rohan Sawhney, Keenan Crane, and Ioannis Gkioulekas. 2023. Boundary Value Caching for Walk on Spheres. *ACM Trans. Graph.* (2023). https://doi.org/10.1145/3592400

Bailey Miller, Rohan Sawhney, Keenan Crane, and Ioannis Gkioulekas. 2024a. Differential Walk on Spheres. *ACM Trans. Graph.* 43, 6, Article 174 (Nov. 2024), 18 pages. https://doi.org/10.1145/3687913

Bailey Miller, Rohan Sawhney, Keenan Crane, and Ioannis Gkioulekas. 2024b. Walkin' Robin: Walk on Stars with Robin Boundary Conditions. *ACM Trans. Graph.* 43, 4 (2024).

Bailey Miller, Rohan Sawhney, Keenan Crane, and Ioannis Gkioulekas. 2025. Solving partial differential equations in participating media. *ACM Trans. Graph.* 44, 4 (july 2025). https://doi.org/10.1145/3731152

Mervin E Muller. 1956. Some continuous Monte Carlo methods for the Dirichlet problem. *The Annals of Mathematical Statistics* 27, 3 (1956), 569–589.

Mohammad Sina Nabizadeh, Ravi Ramamoorthi, and Albert Chern. 2021. Kelvin transformations for simulations on infinite domains. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–15.

Michael Ortner and Lucas Gabriel Coliado Bandeira. 2020. Magpylib: A free Python package for magnetic field computation. *SoftwareX* 11 (2020), 100466. https://doi.org/10.1016/j.softx.2020.100466

Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2023. *Physically based rendering: From theory to implementation.* The MIT Press.

Yang Qi, Dario Seyb, Benedikt Bitterli, and Wojciech Jarosz. 2022. A bidirectional formulation for Walk on Spheres. In *Computer Graphics Forum*, Vol. 41. Wiley Online Library, 51–62.

Damien Rioux-Lavoie, Ryusuke Sugimoto, Tümay Özdemir, Naoharu H. Shimada, Christopher Batty, Derek Nowrouzezahrai, and Toshiya Hachisuka. 2022. A Monte Carlo Method for Fluid Simulation. *ACM Trans. Graph.* 41, 6, Article 240 (nov 2022), 16 pages. https://doi.org/10.1145/3550454.3555450

Rohan Sawhney. 2021. *FCPW: Fastest Closest Points in the West.*

Rohan Sawhney. 2024. *Monte Carlo Geometry Processing: A Grid-Free Approach to Solving Partial Differential Equations on Volumetric Domains.* Ph. D. Dissertation. Carnegie Mellon University, Pittsburgh, PA. CMU-CS-24-125.

Rohan Sawhney and Keenan Crane. 2020. Monte Carlo Geometry Processing: A Grid-Free Approach to PDE-Based Methods on Volumetric Domains. *ACM Trans. Graph.* 39, 4 (2020).

Rohan Sawhney, Bailey Miller, Ioannis Gkioulekas, and Keenan Crane. 2023. Walk on Stars: A Grid-Free Monte Carlo Method for PDEs with Neumann Boundary Conditions. *ACM Trans. Graph.* (2023). https://doi.org/10.1145/3592398

Rohan Sawhney, Bailey Miller, Ioannis Gkioulekas, and Keenan Crane. 2025. State of the Art in Grid-Free Monte Carlo Methods for Partial Differential Equations *(SIGGRAPH Courses '25)*. Association for Computing Machinery, New York, NY, USA, Article 3, 8 pages. https://doi.org/10.1145/3721241.3734001

Rohan Sawhney, Dario Seyb, Wojciech Jarosz, and Keenan Crane. 2022. Grid-Free Monte Carlo for PDEs with Spatially Varying Coefficients. *ACM Trans. Graph.* (2022). https://doi.org/10.1145/3528223.3530134

Hang Si. 2015. TetGen, a Delaunay-Based Quality Tetrahedral Mesh Generator. *ACM transactions on mathematical software* 41, 2 (2015), 1–36.

Ryusuke Sugimoto, Christopher Batty, and Toshiya Hachisuka. 2024a. Velocity-Based Monte Carlo Fluids. In *ACM SIGGRAPH 2024 Conference Papers* (Denver, CO, USA) *(SIGGRAPH '24)*. Association for Computing Machinery, New York, NY, USA, Article 8, 11 pages. https://doi.org/10.1145/3641519.3657405

Ryusuke Sugimoto, Terry Chen, Yiti Jiang, Christopher Batty, and Toshiya Hachisuka. 2023. A Practical Walk-on-Boundary Method for Boundary Value Problems. *ACM Trans. Graph.* 42, 4, Article 81 (jul 2023), 16 pages. https://doi.org/10.1145/3592109

Ryusuke Sugimoto, Nathan King, Toshiya Hachisuka, and Christopher Batty. 2024b. Projected Walk on Spheres: A Monte Carlo Closest Point Method for Surface PDEs. In *ACM SIGGRAPH Asia 2024 Conference Papers* (Tokyo, Japan) *(SIGGRAPH Asia*

'24). Association for Computing Machinery, New York, NY, USA, 10 pages. https://doi.org/10.1145/3680528.3687599
A.N. Tikhonov. 1998. *Nonlinear Ill-posed Problems.* Springer.
Wolfram Research. 2025. *Magnetostatics for Permanent Magnets.* https://reference.wolfram.com/language/PDEModels/tutorial/Electromagnetics/MagnetostaticsForPermanentMagnets.html Wolfram Language & System Documentation Center. Accessed 2025-05-22.
Lifan Wu, Nathan Morrical, Sai Praveen Bangaru, Rohan Sawhney, Shuang Zhao, Chris Wyman, Ravi Ramamoorthi, and Aaron Lefhon. 2025. Unbiased Differential Visibility Using Fixed-Step Walk-on-Spherical-Caps And Closest Silhouettes. *ACM Trans. Graph.* (2025). https://doi.org/10.1145/3731174
Ekrem Fatih Yilmazer, Delio Vicini, and Wenzel Jakob. 2024. Solving Inverse PDE Problems using Monte Carlo Estimators. *Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 43 (Dec. 2024). https://doi.org/10.1145/3687990
Zihan Yu, Lifan Wu, Zhiqian Zhou, and Shuang Zhao. 2024. A Differential Monte Carlo Solver For the Poisson Equation. *ACM Trans. Graph.* 43 (July 2024). https://doi.org/10.1145/3641519.3657460
Ekrem Fatih Yılmazer, Delio Vicini, and Wenzel Jakob. 2022. Solving Inverse PDE Problems using Grid-Free Monte Carlo Estimators. arXiv:2208.02114 [cs.GR]
Yong Zhan, Sanjay V. Kumar, and Sachin S. Sapatnekar. 2008. Thermally Aware Design. *Foundations and Trends® in Electronic Design Automation* 2, 3 (2008), 255–370. https://doi.org/10.1561/1000000007

A DERIVATION OF THE DIRECTIONAL DERIVATIVE BIE FOR THE NEUMANN BOUNDARY

Starting from Equation 8, we derive the integral expression for the directional derivative $\partial_v u$ in Equation 9. The primary challenge lies in eliminating the second-order mixed derivative $\partial^2_{nv} u$ in Equation 8. Our derivation is divided into four steps:

A.1 Normal And Tangential Decomposition

We decompose $\partial^2_{nv} u$ on $\partial \mathrm{St}_N$ into normal and tangential components,

$$\partial^2_{nv} u = \partial^2_n u \, (\boldsymbol{v} \cdot \boldsymbol{n}) + D^2 u \cdot \boldsymbol{n} \cdot \boldsymbol{v}_\Gamma, \tag{30}$$

where D^2 represents the Hessian.

A.2 Rewriting the Tangential Component

We apply the identity [Henrot and Pierre 2018, page 227]

$$D^2 u \cdot \boldsymbol{n} \cdot \boldsymbol{v}_\Gamma = \nabla_\Gamma \partial_n u \cdot \boldsymbol{v}_\Gamma - \nabla u \cdot (\boldsymbol{v} \cdot \nabla_\Gamma \boldsymbol{n}) \tag{31}$$

along with the relations $\boldsymbol{v} \cdot \nabla_\Gamma \boldsymbol{n} = \nabla_\Gamma (\boldsymbol{v} \cdot \boldsymbol{n})$ (as $\boldsymbol{v}$ is constant) and $\partial_n u = h$ (the prescribed Neumann condition). This yields

$$\partial^2_{nv} u = \partial^2_n u \, (\boldsymbol{v} \cdot \boldsymbol{n}) + \nabla_\Gamma h \cdot \boldsymbol{v}_\Gamma - \nabla u \cdot \nabla_\Gamma (\boldsymbol{v} \cdot \boldsymbol{n}). \tag{32}$$

A.3 Rewriting the Normal Component

We apply the Laplacian identity [Henrot and Pierre 2018, Equation 5.59]

$$\Delta u = \Delta_\Gamma u + H \partial_n u + \partial^2_n u = -f, \tag{33}$$

and substitute $\partial^2_n u = -(f + \Delta_\Gamma u + H \partial_n u)$ into Equation 32 to get

$$\partial^2_{nv} u = -(f + \Delta_\Gamma u + Hh)(\boldsymbol{v} \cdot \boldsymbol{n}) + \nabla_\Gamma h \cdot \boldsymbol{v}_\Gamma - \nabla u \cdot \nabla_\Gamma (\boldsymbol{v} \cdot \boldsymbol{n}). \tag{34}$$

A.4 Integration by Parts

Finally, we substitute the right hand side of Equation 34 into the surface integral $\int_{\partial \mathrm{St}_\mathrm{N}} G^\mathrm{B} \partial^2_{nv} u$, and apply integration by parts [Henrot and Pierre 2018, Equation 5.64] to the term involving $\Delta_\Gamma u$:

$$\int_{\partial \mathrm{St}_\mathrm{N}} G^\mathrm{B} (\boldsymbol{v} \cdot \boldsymbol{n}) \Delta_\Gamma u = - \int_{\partial \mathrm{St}_N} \left[(\boldsymbol{v} \cdot \boldsymbol{n}) \nabla_\Gamma G^\mathrm{B} \cdot \nabla u + G^\mathrm{B} \nabla u \cdot \nabla_\Gamma (\boldsymbol{v} \cdot \boldsymbol{n}) \right]. \tag{35}$$

Collecting terms then yields

$$\begin{aligned} &\int_{\partial \mathrm{St}_\mathrm{N}} P^\mathrm{B} \underbrace{\left(\boldsymbol{v}_\Gamma - (\boldsymbol{v} \cdot \boldsymbol{n}) \frac{\nabla_\Gamma G^\mathrm{B}}{P^\mathrm{B}} \right)}_{|\boldsymbol{\rho}_v| \partial_{\boldsymbol{\rho}} u} \cdot \nabla u \\ &+ \int_{\partial \mathrm{St}_\mathrm{N}} P^\mathrm{B} \underbrace{(\boldsymbol{v} \cdot \boldsymbol{n}) h}_{\mu_v} \\ &- \int_{\partial \mathrm{St}_\mathrm{N}} G^\mathrm{B} \underbrace{\left[\partial_{\boldsymbol{v}_\Gamma} h - (\boldsymbol{v} \cdot \boldsymbol{n}) H h - (\boldsymbol{v} \cdot \boldsymbol{n}) f \right]}_{\eta_v}, \end{aligned} \tag{36}$$

which matches all the components of the BIE on $\partial \mathrm{St}_\mathrm{N}$ that we use in the main text.

B DERIVATION OF THE DIRECTIONAL DERIVATIVE BIE FOR THE DIRICHLET BOUNDARY

We derive an analogous BIE to Equation 36, but this time for a star-shaped region St that contains parts of the Dirichlet boundary $\partial \Omega_\mathrm{D}$. As in Appendix A, the challenge lies in handling the second-order mixed derivative $\partial^2_{nv} u$ in the integral

$$\int_{\partial \mathrm{St}_D} G^\mathrm{B} \partial^2_{nv} u. \tag{37}$$

We begin with the identity

$$\partial^2_{nv} u = -(f + \Delta_\Gamma u + H \partial_n u)(\boldsymbol{v} \cdot \boldsymbol{n}) + \nabla_\Gamma \partial_n u \cdot \boldsymbol{v}_\Gamma - \nabla u \cdot \nabla_\Gamma (\boldsymbol{v} \cdot \boldsymbol{n}) \tag{38}$$

from Equation 34, where the normal derivative $\partial_n u$ is no longer known on $\partial \mathrm{St}_\mathrm{D}$. We apply integration by parts to the term involving $\nabla_\Gamma \partial_n u \cdot \boldsymbol{v}_\Gamma$ in the surface integral above to get

$$\int_{\partial \mathrm{St}_\mathrm{D}} G^\mathrm{B} \nabla_\Gamma \partial_n u \cdot \boldsymbol{v}_\Gamma = - \int_{\partial \mathrm{St}_\mathrm{D}} \left[\nabla_\Gamma G^\mathrm{B} \cdot \boldsymbol{v}_\Gamma \partial_n u - G^\mathrm{B} H \boldsymbol{v} \cdot \boldsymbol{n} \partial_n u \right]. \tag{39}$$

Collecting terms, we obtain the final expression for the directional derivative $\partial_v u$ on $\partial \mathrm{St}_\mathrm{D}$:

$$\begin{aligned} \partial_v u = &\int_{\partial \mathrm{St}_\mathrm{D}} P^\mathrm{B} \underbrace{\left((\boldsymbol{v} \cdot \boldsymbol{n}) + \frac{\nabla_\Gamma G^\mathrm{B}}{P^\mathrm{B}} \cdot \boldsymbol{v}_\Gamma \right)}_{|\boldsymbol{\rho}_v|} \partial_n u \\ &+ \int_{\partial \mathrm{St}_\mathrm{D}} P^\mathrm{B} \underbrace{\nabla_\Gamma g \cdot \boldsymbol{v}}_{\mu_v} \\ &- \int_{\partial \mathrm{St}_\mathrm{D}} G^\mathrm{B} \underbrace{\left[-(\boldsymbol{v} \cdot \boldsymbol{n}) \Delta_\Gamma g - \nabla_\Gamma (\boldsymbol{v} \cdot \boldsymbol{n}) \cdot \nabla_\Gamma g - (\boldsymbol{v} \cdot \boldsymbol{n}) f \right]}_{\eta_v}. \end{aligned} \tag{40}$$

C DERIVATION OF THE EDGE INTEGRAL

On polyhedral domains, edges introduce Dirac delta contributions in H and $\partial_{v_\Gamma} h$ within η_v (Equation 12), producing singular terms that must be treated explicitly. We derive Equation 22 by isolating the singular part of the second integral in Equation 9, namely

$$\int_{\partial \mathrm{St}_N} G^\mathrm{B} \left[\partial_{v_\Gamma} h - (\boldsymbol{v} \cdot \boldsymbol{n}) H h \right], \tag{41}$$

where we have omitted the smooth term $-(\boldsymbol{v} \cdot \boldsymbol{n}) f$ for clarity.

C.1 Mollification of Edges

We mollify an edge e (with face normals $\boldsymbol{n}^{\pm}$) by replacing it with a smooth cylindrical surface S_ε of radius $\varepsilon > 0$, yielding a C^2 boundary where $\partial_{\boldsymbol{v}_\Gamma} h - (\boldsymbol{v}\cdot\boldsymbol{n})Hh$ is well-defined and surface calculus applies.

For a constant vector field $\boldsymbol{v}$, we can employ the surface divergence identity [Henrot and Pierre 2018, Equation 5.55]:

$$\nabla_\Gamma \cdot \boldsymbol{v}_\Gamma = -(\boldsymbol{v}\cdot\boldsymbol{n})H, \tag{42}$$

where $\boldsymbol{v}_\Gamma = \boldsymbol{v} - (\boldsymbol{v}\cdot\boldsymbol{n})\boldsymbol{n}$ is the tangential component of $\boldsymbol{v}$ on the surface. Multiplying this identity by the Neumann data h and rearranging, we obtain

$$(\boldsymbol{v}\cdot\boldsymbol{n})Hh = -\nabla_\Gamma\cdot(h\boldsymbol{v}_\Gamma) + \nabla_\Gamma h\cdot\boldsymbol{v}_\Gamma. \tag{43}$$

Using $\partial_{\boldsymbol{v}_\Gamma} h = \nabla_\Gamma h\cdot\boldsymbol{v}_\Gamma$, the terms cancel in Equation 41, leaving

$$\int_{S_\varepsilon} G^{\mathrm{B}}\nabla_\Gamma\cdot(h\boldsymbol{v}_\Gamma)\,\mathrm{d}A, \tag{44}$$

C.2 Surface Divergence Theorem

Applying the surface divergence theorem to Equation 44 gives

$$\int_{S_\varepsilon} G^{\mathrm{B}}\nabla_\Gamma\cdot(h\boldsymbol{v}_\Gamma)\mathrm{d}A = \oint_{\partial S_\varepsilon} G^{\mathrm{B}} h(\boldsymbol{v}_\Gamma\cdot\boldsymbol{t})\mathrm{d}l, \tag{45}$$

where $\boldsymbol{t}$ is the unit tangent vector to ∂S_ε.

The boundary ∂S_ε has two circular arcs (radius ε) and two segments on the faces. As $\varepsilon\to 0$, the arcs vanish and the segments collapse to the edge e. On them, $\boldsymbol{v}_\Gamma\cdot\boldsymbol{t} = \boldsymbol{v}\cdot\boldsymbol{t}^{\pm}$; taking the limit (allowing different Neumann data on each side) yields

$$\int_e G^{\mathrm{B}}\left[(\boldsymbol{v}\cdot\boldsymbol{t}^{+})h^{+} + (\boldsymbol{v}\cdot\boldsymbol{t}^{-})h^{-}\right]\mathrm{d}l. \tag{46}$$

Summing over all edges gives the edge integral in Equation 22.

D MITIGATING SINGULAR KERNELS WITH CONTROL VARIATES

Monte Carlo estimators of boundary integrals can have high variance when kernels are singular near x. This affects the Neumann boundary integral (Equation 3), the edge integral (Equation 22), and the second-order normal derivative (Equation 16). We reduce variance with control variates: subtract a locally matching singular surrogate and add back an equivalent smooth integral.

D.1 Neumann Boundary Integral

The Green's function in $\int_{\partial\mathrm{St}_N} G^{\mathrm{B}}(x,z)\eta_v(z)\mathrm{d}z$ is singular as $z\to x$. To address it, we introduce a control variate based on a constant vector field $\hat{\boldsymbol{u}}$. We define this field such that its normal component matches the singular term: $\hat{\boldsymbol{u}}\cdot\boldsymbol{n}(\bar{x}) = \eta_v(\bar{x})$, where $\bar{x}$ is the point on $\partial\mathrm{St}_N$ closest to x. It induces a linear potential function $\phi(z) = \hat{\boldsymbol{u}}\cdot z$, which satisfies Laplace's equation. Therefore, ϕ satisfies the boundary integral equation from Equation 3:

$$\phi(x) = \int_{\partial\mathrm{St}} P^{\mathrm{B}}(x,z)\phi(z)\mathrm{d}z - \int_{\partial\mathrm{St}} G^{\mathrm{B}}(x,z)(\hat{\boldsymbol{u}}\cdot\boldsymbol{n}(z))\mathrm{d}z.$$

Using $\int_{\partial\mathrm{St}} P^{\mathrm{B}}(x,z)\mathrm{d}z = 1$ gives the identity:

$$\int_{\partial\mathrm{St}} G^{\mathrm{B}}(x,z)(\hat{\boldsymbol{u}}\cdot\boldsymbol{n}(z))\mathrm{d}z = \int_{\partial\mathrm{St}} P^{\mathrm{B}}(x,z)\hat{\boldsymbol{u}}\cdot(z-x)\mathrm{d}z. \tag{47}$$

Subtract $\int_{\partial\mathrm{St}_N} G^{\mathrm{B}}(x,z)(\hat{\boldsymbol{u}}\cdot\boldsymbol{n}(z))\mathrm{d}z$ and add back its equivalent from Equation 47:

$$\begin{aligned}\int_{\partial\mathrm{St}_N} G^{\mathrm{B}}(x,z)\eta_v(z)\mathrm{d}z = &\int_{\partial\mathrm{St}_N} G^{\mathrm{B}}(x,z)(\eta_v(z) - \hat{\boldsymbol{u}}\cdot\boldsymbol{n}(z))\mathrm{d}z\\ &+ \int_{\partial\mathrm{St}} P^{\mathrm{B}}(x,z)\hat{\boldsymbol{u}}\cdot(z-x)\mathrm{d}z.\end{aligned} \tag{48}$$

Then $(\eta_v - \hat{\boldsymbol{u}}\cdot\boldsymbol{n})\to 0$ as $z\to\bar{x}$, which reduces the impact of the singularity of G^{B}, and P^{B} can be importance sampled.

D.2 Edge Integral

The edge integral in Equation 22 also suffers from high variance due to the singularity of $G^{\mathrm{B}}(x,z)$. We introduce a control variate using a constant vector field $\hat{\boldsymbol{u}}$ chosen such that $\hat{\boldsymbol{u}}\cdot\boldsymbol{n}^{\pm}(\bar{x}) = h^{\pm}(\bar{x})$, where $h^{\pm}(\bar{x})$ are the prescribed Neumann data on the faces adjacent to the edge, and $\bar{x}$ is the point on the edge $\partial\mathrm{St}^{\mathrm{E}}_{\mathrm{N}}$ closest to x. This involves subtracting and adding the term:

$$\int_{\partial\mathrm{St}^{\mathrm{E}}_{\mathrm{N}}} G^{\mathrm{B}}(x,z)\left[(\hat{\boldsymbol{u}}\cdot\boldsymbol{n}^{-}(z))(\boldsymbol{v}\cdot\boldsymbol{t}^{-}(z)) + (\hat{\boldsymbol{u}}\cdot\boldsymbol{n}^{+}(z))(\boldsymbol{v}\cdot\boldsymbol{t}^{+}(z))\right]\mathrm{d}l. \tag{49}$$

Transform this using integration by parts for $\phi(z) = \hat{\boldsymbol{u}}\cdot z$ on $\partial\mathrm{St}_{\mathrm{N}}$:

$$\begin{aligned}&\int_{\partial\mathrm{St}^{\mathrm{E}}_{\mathrm{N}}} G^{\mathrm{B}}(x,z)\sum_{\pm}(\hat{\boldsymbol{u}}\cdot\boldsymbol{n}^{\pm}(z))(\boldsymbol{v}\cdot\boldsymbol{t}^{\pm}(z))\mathrm{d}z\\ &= \int_{\partial\mathrm{St}_{\mathrm{N}}}(\boldsymbol{v}\cdot\boldsymbol{n}(z))(\nabla_\Gamma G(x,z)\cdot\hat{\boldsymbol{u}})\mathrm{d}z.\end{aligned} \tag{50}$$

The edge integral becomes:

$$\begin{aligned}&\int_{\partial\mathrm{St}^{\mathrm{E}}_{\mathrm{N}}} G^{\mathrm{B}}(x,z)\sum_{\pm}\left[(h^{\pm}(z) - \hat{\boldsymbol{u}}\cdot\boldsymbol{n}^{\pm}(z))(\boldsymbol{v}\cdot\boldsymbol{t}^{\pm}(z))\right]\mathrm{d}z\\ &+ \int_{\partial\mathrm{St}_{\mathrm{N}}}(\boldsymbol{v}\cdot\boldsymbol{n}(z))(\nabla_\Gamma G(x,z)\cdot\hat{\boldsymbol{u}})\mathrm{d}z.\end{aligned} \tag{51}$$

Now $(h^{\pm} - \hat{\boldsymbol{u}}\cdot\boldsymbol{n}^{\pm})\to 0$ near $\bar{x}$, regularizing the singularity and the surface term can reuse the direction samples from the primal walk.

D.3 Second-Order Normal Derivative Integral

Estimating $\partial_n^2 u(x)$ also suffers from singular $\partial_{n_x} P^{\mathrm{B}}$ and $\partial_{n_x} G^{\mathrm{B}}$. Following Yu et al. [2024, Eq. 26], we use control variates from $h(x)$ and $\partial_{n_x} f(x)$:

$$\begin{aligned}&\int_{\partial\mathrm{B}(c,R)}(\partial_{n_x}u(z) - h(x))\partial_{n_x}P^{\mathrm{B}}(x,z)\mathrm{d}z\\ &+\int_{\mathrm{B}(c,R)}(\partial_{n_x}f(y) - \partial_{n_x}f(x))\partial_{n_x}G^{\mathrm{B}}(x,y)\mathrm{d}y + \partial_{n_x}f(x)\frac{R}{N}\end{aligned} \tag{52}$$

Here $\mathrm{B}(c,R)$ is the off-centered ball with kernels P^{B}, G^{B} and N the dimension. The differences $(\partial_{n_x}u(z) - h(x))$ and $(\partial_{n_x}f(y) - \partial_{n_x}f(x))$ vanish as the point approaches x, alleviating the singularities and improving the robustness of the derivative estimator for Neumann shape optimization.