# Monte Carlo Rendering of Biharmonic Diffusion Curves

PAUL HIMMLER and TOBIAS GÜNTHER, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

(a) Input geometry (b) Rendering of biharmonic diffusion curves (ours) (c) Rendering of regular diffusion curves

Fig. 1. Biharmonic diffusion curves are smooth vector graphics primitives that prescribe color and normal derivative on each curve boundary point. We introduce the first Monte Carlo method for the rendering of biharmonic diffusion curves, which requires solving an inverse problem. The input geometry is shown in (a). Starting with an initial guess of the unknown Dirichlet boundary data of the underlying Laplace problem, we perform a gradient-based optimization to meet all boundary conditions (b). In comparison, regular diffusion curves cannot enforce Dirichlet *and* Neumann data at the same time (c).

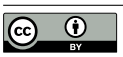
Stochastic Monte Carlo solvers for partial differential equations (PDEs) recently gained popularity in computer graphics, finding applications in geometry processing, rendering, simulation, and visualization. At present, there exists no Monte Carlo solver for the rendering of biharmonic diffusion curves, an artist-friendly smooth vector graphics primitive. The fourth-order biharmonic equation of biharmonic diffusion curves can be split into two second-order PDEs, namely a Laplace and a Poisson equation. However, since biharmonic diffusion curves set Dirichlet and inhomogeneous Neumann conditions at the same time, these two second-order PDEs are tightly coupled and can hence not be solved directly. We propose to treat the rendering of biharmonic diffusion curves as an inverse problem, in which the Dirichlet data of the Laplace equation is unknown. We formulate a variational energy optimization, such that the user-defined boundary conditions are met. Thereby, the necessary gradients are estimated stochastically by solving two second-order problems with Dirichlet boundary conditions only.

CCS Concepts: • **Computing methodologies** → **Rendering**.

Additional Key Words and Phrases: Biharmonic, Monte Carlo, Partial Differential Equations, Walk-on-Spheres

**ACM Reference Format:**
Paul Himmler and Tobias Günther. 2026. Monte Carlo Rendering of Biharmonic Diffusion Curves. *ACM Trans. Graph.* 45, 4, Article 107 (July 2026), 12 pages. https://doi.org/10.1145/3811397

Authors' Contact Information: Paul Himmler, paul.himmler@fau.de; Tobias Günther, tobias.guenther@fau.de, Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Bavaria, Germany.




ACM 1557-7368/2026/7-ART107
https://doi.org/10.1145/3811397

## 1 Introduction

In the field of smooth vector graphics [Tian and Günther 2024], diffusion curves [Orzan et al. 2008] define an image through a set of curves along which a color is specified on both sides. These user-defined colors are diffused into the image domain to produce a smoothly shaded and resolution-independent image that is often times more compact than a discrete raster graphic. Formally, the equilibrium of the diffusion process is found by solving a Laplace equation, with the curve colors modeled as Dirichlet boundary conditions [Jeschke et al. 2009]. This treats the image generation as a smooth boundary value interpolation problem. The approach was inspired by gradient-domain painting [McCann and Pollard 2008] and numerous follow-up works concentrated on efficient rendering [Bang et al. 2023; Bowers et al. 2011; Pang et al. 2011; Prévost et al. 2015; Sun et al. 2014, 2012] and vectorization [Lu et al. 2019; Zhao et al. 2017]. Instead of prescribing the color on the boundary with a Dirichlet boundary condition, one may alternatively prescribe the color gradient on the boundary via a Neumann boundary condition [Bang et al. 2023; Tian and Günther 2025]. More generally, linear combinations of color and color gradient can be imposed using Robin boundary conditions [Miller et al. 2024b]. However, it is distinctly *not* possible to define a color *and* a color gradient at the same time, since *two* boundary conditions over-constrain the second-order Laplace equation. To overcome this problem *biharmonic diffusion curves* [Boyé et al. 2012; Finch et al. 2011; Ilbery et al. 2013] replace the Laplace equation with a fourth-order biharmonic equation, which allows for finding a smooth image that meets both boundary constraints. This provides artistic control that is not available with regular diffusion curves [Orzan et al. 2008].

Recently, Sawhney and Crane [2020] introduced the Walk-on-Spheres (WoS) method [Muller 1956] to the graphics community, which solves many commonly-used elliptic PDEs in a grid-free and

point-wise manner. Their work also discussed the generalization of the WoS method to biharmonic equations under the assumption that both the color and its *Laplacian* are known on the boundary. In this special case, the biharmonic equation can be split into two second-order PDEs with well-defined Dirichlet boundary data, allowing them to be solved in a nested manner. In contrast, biharmonic diffusion curves [Boyé et al. 2012; Ilbery et al. 2013] prescribe the color and its *normal derivative* instead. This turns out to be more complicated to solve, since the common split of the fourth-order biharmonic equation into two second-order PDEs results in a coupled system, in which one PDE lacks boundary data. Despite extensive research in mathematics on WoS [Sabelfeld 1991], including solutions for biharmonic equations with prescribed Laplacian on the boundary [Sabelfeld and Shkarupa 2003], so far, no Monte Carlo method exists for biharmonic equations when the Laplacian of the unknown is not already given on the boundary, i.e., there is no readily-available WoS variant for biharmonic diffusion curves.

In this paper, we develop the first Monte Carlo solver for biharmonic diffusion curves [Boyé et al. 2012; Ilbery et al. 2013]. We split the fourth-order biharmonic equation into two *coupled* linear elliptic problems: a Laplace and a Poisson problem. The Poisson problem is over-constrained on its own, but the additional degree of freedom is absorbed by the unknown Dirichlet boundary data of the Laplace problem. Rendering a biharmonic diffusion curve image thereby becomes an *inverse problem*, in which the unknown Dirichlet boundary data of the Laplace problem has to be found such that the user-defined color and color gradient constraints are satisfied. We formulate this inverse problem as variational energy minimization, and accelerate the gradient calculation using reservoir sampling [Vitter 1985], mean value caching [Bakbouk and Peers 2023], and path replay [Yilmazer et al. 2024]. Fig. 1 gives a comparison with regular diffusion curves. In summary, we contribute:

- a split formulation for *biharmonic diffusion curves* into two coupled linear elliptic equations,
- a variational energy minimization for the unknown Dirichlet boundary data of the Laplace equation,
- a Monte Carlo estimator for the gradients with respect to the unknown Dirichlet boundary data,
- and lastly, a demonstration of acceleration strategies, including reservoir sampling and mean value caching.

## 2 Related Work

### 2.1 Poisson Equations

The Poisson equation is a second-order linear elliptic partial differential equation (PDE) that frequently appears in many branches of graphics [Esturo et al. 2013; Hou et al. 2018; Yu et al. 2004]. The unknown in this equation is the function $\mathbf{u}(\mathbf{x}) : \Omega \to \mathbb{R}^n$, where $\mathbf{v}(\mathbf{x}) : \Omega \to \mathbb{R}^n$ is a given *source term*. For the remainder of this paper, we are only interested in Poisson equations with Dirichlet boundary conditions $\mathbf{f}(\mathbf{x}) : \partial\Omega \to \mathbb{R}^n$, i.e.:

$$\Delta \mathbf{u}(\mathbf{x}) = \mathbf{v}(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \Omega, \tag{1}$$

$$\text{Dirichlet:} \qquad \mathbf{u}(\mathbf{x}) = \mathbf{f}(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \partial\Omega, \tag{2}$$

where $\Delta$ is the negative-semidefinite Laplace operator. Other types of boundary conditions include Neumann and Robin boundary conditions [Miller et al. 2024b; Sawhney et al. 2023], which are not required in our approach. An important special case is the *Laplace equation*, for which the source term vanishes, i.e., $\mathbf{v}(\mathbf{x}) = \mathbf{0}$. In curve-based smooth vector graphics, the Laplace equation is used for diffusion curves [Jeschke et al. 2009], which model the image synthesis as smooth boundary value interpolation problem. Note that Poisson equations can be solved component-wise. To later optimize over all components at once, we utilize a vector-valued notation.

*Boundary Integral Equation.* The entry point for Monte Carlo methods is a formulation in terms of integral equations [Sawhney and Crane 2020]. For the Poisson equation in (1), a direct boundary integral equation exists [Costabel 1987] for any two subsets of $\mathcal{A}, C \subset \Omega$ of the domain $\Omega$, cf. Sawhney et al. [2023]:

$$\begin{aligned} a(\mathbf{x})\,\mathbf{u}(\mathbf{x}) = &\int_{\partial\mathcal{A}} \mathcal{P}^C(\mathbf{x},\mathbf{z})\,\mathbf{u}(\mathbf{z}) - \mathcal{G}^C(\mathbf{x},\mathbf{z})\frac{\partial \mathbf{u}(\mathbf{z})}{\partial \mathbf{n}_\mathbf{z}}\,\mathrm{d}\mathbf{z} \\ &+ \int_{\mathcal{A}} \mathcal{G}^C(\mathbf{x},\mathbf{y})\,\mathbf{v}(\mathbf{y})\,\mathrm{d}\mathbf{y}, \end{aligned} \tag{3}$$

where $a(\mathbf{x}) = 1$ in the interior ($\forall \mathbf{x} \in \mathcal{A}$) and $a(\mathbf{x}) = 1/2$ on the boundary ($\forall \mathbf{x} \in \partial\mathcal{A}$). Thereby, $\mathcal{P}^C = \partial\mathcal{G}^C/\partial\mathbf{n}_\mathbf{z}$ is the Poisson kernel, which is the normal derivative of the scalar-valued Green's function $\mathcal{G}^C$, wrt. the inward pointing normal $\mathbf{n}_\mathbf{z}$. Later, we also use $\partial_\mathbf{n}\mathbf{u}(\mathbf{z}) := \frac{\partial \mathbf{u}(\mathbf{z})}{\partial \mathbf{n}_\mathbf{z}}$ to denote the normal derivative. $\mathcal{G}^C$ and $\mathcal{P}^C$ are known in closed-form only for few domains, e.g., for the 2D ball.

*Special case: 2D Ball.* If $\mathcal{A}, C$ are equal to a ball $\mathcal{B}(\mathbf{x})$ centered at $\mathbf{x}$ with radius $R$, then Eq. (3) simplifies with $\mathcal{A} = C = \mathcal{B}(\mathbf{x})$ if $\mathbf{u}(\mathbf{x})$ is evaluated at the interior center point $\mathbf{x}$ itself:

$$\begin{aligned} \mathbf{u}(\mathbf{x}) = &\int_{\partial\mathcal{B}(\mathbf{x})} \mathcal{P}^\mathcal{B}(\mathbf{x},\mathbf{z})\,\mathbf{u}(\mathbf{z})\,\mathrm{d}\mathbf{z} \\ &+ \int_{\mathcal{B}(\mathbf{x})} \mathcal{G}^\mathcal{B}(\mathbf{x},\mathbf{y})\,\mathbf{v}(\mathbf{y})\,\mathrm{d}\mathbf{y}. \end{aligned} \tag{4}$$

Further, for a 2D ball, the local Green's function has a closed-form expression for any interior point $\mathbf{y} \in \mathcal{B}(\mathbf{x})$ with $r = \|\mathbf{x} - \mathbf{y}\|$:

$$\mathcal{G}^\mathcal{B}(\mathbf{x},\mathbf{y}) = \frac{1}{2\pi}\log\left(\frac{R}{r}\right). \tag{5}$$

Similarly, the local Poisson kernel has a radially-symmetric closed-form solution on the boundary $\mathbf{z} \in \partial\mathcal{B}(\mathbf{x})$ of the ball:

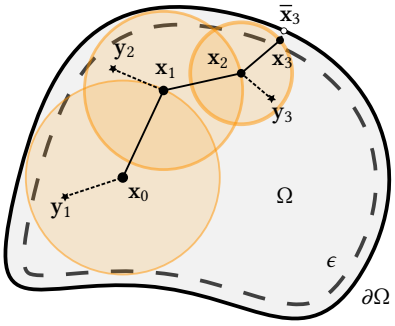
$$\mathcal{P}^\mathcal{B}(\mathbf{x},\mathbf{z}) = \frac{\partial \mathcal{G}^\mathcal{B}(\mathbf{x},\mathbf{z})}{\partial \mathbf{n}_\mathbf{z}} = \frac{1}{2\pi R}. \tag{6}$$

*Walk-on-Spheres.* For a given point $\mathbf{x}_k \in \Omega$, the WoS estimator [Muller 1956] numerically integrates Eq. (4) by recursively sampling the source contribution from an interior point $\mathbf{y}_{k+1} \in \mathcal{B}(\mathbf{x}_k)$ inside the largest possible ball $\mathcal{B}(\mathbf{x}_k)$ around $\mathbf{x}_k$ and by moving to the next boundary point $\mathbf{x}_{k+1} \in \partial\mathcal{B}(\mathbf{x}_k)$ on the ball:

$$\widehat{\mathbf{u}}(\mathbf{x}_k) := \frac{\mathcal{P}^\mathcal{B}(\mathbf{x}_k,\mathbf{x}_{k+1})\,\widehat{\mathbf{u}}(\mathbf{x}_{k+1})}{p^{\partial\mathcal{B}}(\mathbf{x}_{k+1})} + \frac{\mathcal{G}^B(\mathbf{x}_k,\mathbf{y}_{k+1})\,\mathbf{v}(\mathbf{y}_{k+1})}{p^\mathcal{B}(\mathbf{y}_{k+1})}. \tag{7}$$

Thereby, $\mathbf{x}_{k+1}$ and $\mathbf{y}_{k+1}$ are sampled according to the probability distributions $p^{\partial\mathcal{B}}(\mathbf{x}_{k+1})$ and $p^\mathcal{B}(\mathbf{y}_{k+1})$, respectively. The value $\widehat{\mathbf{u}}(\mathbf{x}_{k+1})$

is estimated recursively from either a single or multiple samples. Eventually, the recursion stops, once the next sample point $\mathbf{x}_{k+1}$ falls within an $\epsilon$-shell around the boundary $\partial\Omega$, where the Dirichlet boundary condition of the closest boundary point $\overline{\mathbf{x}}_{k+1}$ is evaluated, i.e., $\widehat{\mathbf{u}}(\mathbf{x}_{k+1}) := \mathbf{f}(\overline{\mathbf{x}}_{k+1})$.




The snapping to the boundary is necessary to terminate. The induced bias can be removed in 2D for piecewise linear boundary geometry using conformal maps [Himmler and Günther 2025]. Averaging the contributions of multiple such random walks reduces the noise in the Monte Carlo estimate.

*Further Solvers.* Sawhney et al. [2022] extended WoS to PDEs with spatially-varying coefficients, and supported Neumann conditions without walk reflections [Sawhney et al. 2023], while Miller et al. [2024b] generalized WoS to Robin boundary conditions. Driven by experience in Monte Carlo rendering, the visual computing community has quickly embraced and extended stochastic Monte Carlo solvers, e.g., with variance reduction using mean value caching [Bakbouk and Peers 2023], boundary value caching [Miller et al. 2023], neural caching [Li et al. 2023], and harmonic caching [Zhou et al. 2025]. Variance was reduced with off-centered walks [Hwang et al. 2015], bi-directional path construction [Qi et al. 2022], utilizing the continuity of Brownian motion paths [Czekanski et al. 2024], using conformal maps to semicircle and circle segments [Himmler and Günther 2025], path guiding [Huang et al. 2025], and adequate weighting of off-centered samples [Bao et al. 2025]. Inverse problems became solvable with differentiable solvers [Miller et al. 2024a; Yilmazer et al. 2024; Yu et al. 2025, 2024]. Monte Carlo methods have been employed for flow visualization [Tian and Günther 2025], and for fluid simulation in both vorticity [Rioux-Lavoie et al. 2022] and velocity representation [Sugimoto et al. 2024]. The latter made use of the Walk-on-Boundary (WoB) method [Sabelfeld 1982; Sabelfeld and Simonov 1994; Sugimoto et al. 2023], an alternative boundary-only Monte Carlo approach derived from potential theory. Its discrete counterpart includes the method of fundamental solutions and boundary element methods, which have likewise seen strong improvements in recent years [Chen et al. 2024, 2025].

## 2.2 Biharmonic Equations

Biharmonic diffusion curves [Boyé et al. 2012; Ilbery et al. 2013] offer even more artistic freedom than diffusion curves. Formally, they are based on a biharmonic equation, which is a fourth-order partial differential equation in the unknown field $\mathbf{u}(\mathbf{x}) : \Omega \to \mathbb{R}^n$:

$$\Delta^2\mathbf{u}(\mathbf{x}) = \mathbf{0} \qquad \text{for } \mathbf{x} \in \Omega. \tag{8}$$

While second-order PDEs as in Eq. (1) require *one* boundary condition per point, a biharmonic equation as in Eq. (8) needs *two* independent boundary conditions to form a well-defined boundary value problem [Ilbery et al. 2013]. Typical boundary quantities include the value $\mathbf{u}(\mathbf{x})$ (Dirichlet), the normal derivative $\partial_{\mathbf{n}}\mathbf{u}(\mathbf{x})$ (Neumann), the Laplacian $\Delta\mathbf{u}(\mathbf{x})$ (bending moment), and its normal derivative $\partial_{\mathbf{n}}\Delta\mathbf{u}(\mathbf{x})$ (shear force). Sawhney and Crane [2020] consider the case where the value and the Laplacian are prescribed on the boundary $\partial\Omega$ with $\mathbf{f}(\mathbf{x}) : \partial\Omega \to \mathbb{R}^n$ and $\mathbf{h}(\mathbf{x}) : \partial\Omega \to \mathbb{R}^n$:

$$\mathbf{u}(\mathbf{x}) = \mathbf{f}(\mathbf{x}), \quad \Delta\mathbf{u}(\mathbf{x}) = \mathbf{h}(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \partial\Omega. \tag{9}$$

Here, the fourth-order PDE can be split into two second-order PDEs with fully specified Dirichlet boundary data, allowing both to be solved with two nested WoS solvers. Biharmonic diffusion curves [Boyé et al. 2012; Ilbery et al. 2013], on the other hand, prescribe the value and the normal derivative [Scholz 1978] using $\mathbf{f}(\mathbf{x}) : \partial\Omega \to \mathbb{R}^n$ and $\mathbf{g}(\mathbf{x}) : \partial\Omega \to \mathbb{R}^n$:

$$\mathbf{u}(\mathbf{x}) = \mathbf{f}(\mathbf{x}), \quad \partial_{\mathbf{n}}\mathbf{u}(\mathbf{x}) = \mathbf{g}(\mathbf{x}). \qquad \text{for } \mathbf{x} \in \partial\Omega. \tag{10}$$

Unfortunately, it is no longer possible to directly solve this equation with two nested WoS walks, since the Dirichlet data of the Laplace problem is no longer given. This is the setting we treat in this paper. Previously, Finch et al. [2011] imposed a weighted combination of Laplacian and bi-Laplacian constraints (thin-plate splines) on a regular grid in a least-squares manner, offering control over color values and gradients. Boyé et al. [2012] used the finite element method (FEM) to solve Eq. (8). To obtain $C^0$ continuous quadratic patches they used third-order non-conforming elements [De Veubeke 1974]. Moving curves interactively, however, requires remeshing of the domain. Ilbery et al. [2013] used a boundary element method (BEM) instead, in which the image is described as weighted sum of four orthogonal kernel functions defined per line segment. The kernel weights are solved with a dense linear system to meet the color and color gradient constraints. In contrast, our Monte Carlo split formulation solves only second-order PDEs, requires no mesh in the domain interior, assembles no global matrices, and uses embarrassingly parallel random walk evaluations. Outside of vector graphics, biharmonic equations and energy terms have been widely used in mesh deformation [Jacobson et al. 2011; Liu et al. 2025; Thiery et al. 2024; Weber et al. 2012], where boundary constraints are also imposed implicitly as part of a variational minimization. Up until now, no Monte Carlo approach exists for biharmonic diffusion curves.

# 3 Monte Carlo Method for Biharmonic Diffusion Curves

## 3.1 Problem Statement

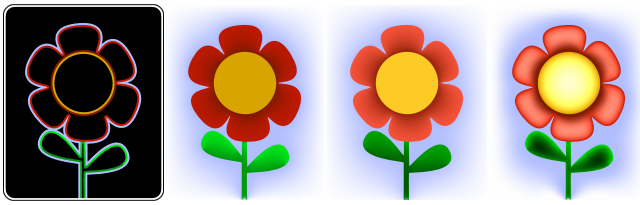
Unlike regular diffusion curves with Dirichlet or Robin boundaries, biharmonic diffusion curves prescribe on each boundary point both a color value (Dirichlet condition) and its normal derivative (Neumann condition) [Boyé et al. 2012; Ilbery et al. 2013], see Fig. 2. To meet both constraints exactly, biharmonic diffusion curves enforce higher-order smoothness in the interior by means of a biharmonic equation. Formally, the biharmonic equation is a fourth-order PDE in the unknown field $\mathbf{u}(\mathbf{x}) : \Omega \to \mathbb{R}^n$:

$$\Delta^2\mathbf{u}(\mathbf{x}) = \mathbf{0} \qquad \text{for } \mathbf{x} \in \Omega, \tag{11}$$

$$\text{Dirichlet:} \quad \mathbf{u}(\mathbf{x}) = \mathbf{f}(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \partial\Omega, \tag{12}$$

$$\text{Neumann:} \quad \partial_{\mathbf{n}}\mathbf{u}(\mathbf{x}) = \mathbf{g}(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \partial\Omega, \tag{13}$$

where biharmonic diffusion curves are treated as part of the domain boundary $\partial\Omega$. While the biharmonic equation can be solved with



(a) Input geometry (b) Diffusion curves (c) Robin $\mu = 33$ [2024b] (d) Biharmonic curves

Fig. 2. Regular diffusion curves prescribe only color values along curves. Robin boundaries ($\partial_{\mathbf{n}}\mathbf{u} - \mu\mathbf{u}$) [2024b] enforce a linear combination of color and normal derivative, meeting neither exactly. Biharmonic diffusion curves meet both color and normal derivative precisely, offering more control.

FEM [Boyé et al. 2012] and BEM [Ilbery et al. 2013], we instead develop a stochastic method. We rewrite the fourth-order PDE in (11) as two coupled second-order PDEs by introducing the auxiliary field $\mathbf{v}(\mathbf{x}) = \Delta\mathbf{u}(\mathbf{x})$. This yields a Laplace equation for $\mathbf{v}(\mathbf{x})$,

$$\Delta\mathbf{v}(\mathbf{x}) = \mathbf{0} \qquad \text{for } \mathbf{x} \in \Omega, \tag{14}$$

$$\text{Dirichlet:} \quad \mathbf{v}(\mathbf{x}) = \mathbf{h}(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \partial\Omega, \tag{15}$$

and a coupled Poisson equation for $\mathbf{u}(\mathbf{x})$,

$$\Delta\mathbf{u}(\mathbf{x}) = \mathbf{v}(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \Omega, \tag{16}$$

$$\text{Dirichlet:} \quad \mathbf{u}(\mathbf{x}) = \mathbf{f}(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \partial\Omega, \tag{17}$$

$$\text{Neumann:} \quad \partial_{\mathbf{n}}\mathbf{u}(\mathbf{x}) = \mathbf{g}(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \partial\Omega. \tag{18}$$

When a source term is given, the boundary conditions in Eqs. (17) and (18) over-constrain the Poisson equation (16), since a second-order PDE normally requires either a Dirichlet *or* a Neumann condition, but not both. However, the coupling with Eq. (14) via the source term in Eq. (16) introduces an additional degree of freedom through the unknown boundary function $\mathbf{h}(\mathbf{x})$ in Eq. (15), which absorbs this over-constraint. In principle, one could impose a Neumann condition on $\partial_{\mathbf{n}}\mathbf{v}(\mathbf{x})$ instead of a Dirichlet condition on $\mathbf{v}(\mathbf{x})$, which we comment on later in the conclusion. Regardless of the chosen form, the task of our stochastic optimization method is to find a boundary function $\mathbf{h}(\mathbf{x})$ for $\mathbf{x} \in \partial\Omega$ that determines the unknown source term $\mathbf{v}(\mathbf{x})$ such that the coupled system reproduces the desired biharmonic solution. In our formulation, the user can also prescribe the Laplacian via $\Delta\mathbf{u} = \mathbf{h}(\mathbf{x})$ (for $\mathbf{x} \in \partial\Omega$) instead of the normal derivative via Eq. (18). At those selected places, $\mathbf{h}(\mathbf{x})$ is simply given and not solved for.

## 3.2 Variational Formulation

The split system in Eqs. (14)–(18) shows that the biharmonic problem becomes fully determined once the boundary function $\mathbf{h}(\mathbf{x})$ in Eq. (15) is specified. For any candidate $\mathbf{h}$, the Laplace equation (14) produces a unique harmonic field $\mathbf{v}_\mathbf{h}$ in the interior, and the Poisson equation (16) produces a unique field $\mathbf{u}_\mathbf{h}$ satisfying the Dirichlet condition in Eq. (17). Thus, the pair $(\mathbf{u}_\mathbf{h}, \mathbf{v}_\mathbf{h})$ is uniquely determined by $\mathbf{h}$. Importantly, both problems are second-order elliptic PDEs with Dirichlet boundary data and can therefore be evaluated using standard Walk-on-Spheres [Muller 1956].

Our goal is to determine the particular boundary function $\mathbf{h}^\star$ for which the induced field $\mathbf{u}_{\mathbf{h}^\star}$ satisfies not only the prescribed Dirichlet condition but also the Neumann condition (18). Since the equations $\Delta\mathbf{v}_\mathbf{h} = \mathbf{0}$ and $\Delta\mathbf{u}_\mathbf{h} = \mathbf{v}_\mathbf{h}$ are enforced exactly by the two Dirichlet problems, the only remaining constraint that depends on $\mathbf{h}$ is the Neumann boundary condition. This observation allows us to formulate an energy functional whose minimizer recovers $\mathbf{h}^\star$.

*Functional.* We measure the deviation from the prescribed Neumann data using the functional $\mathcal{F} : \{\mathbf{h} : \partial\Omega \to \mathbb{R}^n\} \to \mathbb{R}$:

$$\mathcal{F}[\mathbf{h}] = \frac{1}{2}\int_{\partial\Omega} \left\|\partial_{\mathbf{n}}\mathbf{u}_\mathbf{h}(\mathbf{x}) - \mathbf{g}(\mathbf{x})\right\|^2 \mathrm{d}S(\mathbf{x}). \tag{19}$$

Since the Dirichlet constraints are satisfied by construction, this functional is sufficient to characterize the correct solution of the biharmonic problem. The optimal boundary data $\mathbf{h}^*$ is defined as

$$\mathbf{h}^* = \arg\min_{\mathbf{h}} \mathcal{F}[\mathbf{h}]. \tag{20}$$

Because the biharmonic boundary value problem (11)–(13) has a unique solution [Ilbery et al. 2013], there exists exactly one boundary function $\mathbf{h}^*$ that reproduces it in the split formulation. Because the functional $\mathcal{F}$ is strictly convex in $\mathbf{h}$, this $\mathbf{h}^*$ is a unique minimizer.

*Euler–Lagrange Condition.* To characterize the minimizer, we consider a variation $\mathbf{h}_\varepsilon = \mathbf{h} + \varepsilon\,\eta$ with $\eta : \partial\Omega \to \mathbb{R}^n$ and compute the first variation of $\mathcal{F}$:

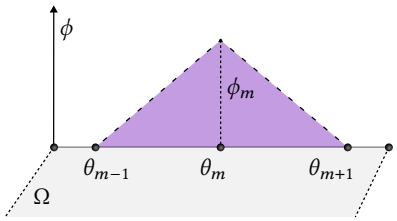
$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\mathcal{F}[\mathbf{h}_\varepsilon]\Big|_{\varepsilon=0} = \int_{\partial\Omega} \frac{\delta\mathcal{F}}{\delta\mathbf{h}}(\mathbf{x}) \cdot \eta(\mathbf{x})\,\mathrm{d}S(\mathbf{x}). \tag{21}$$

A necessary condition for optimality is [Gelfand and Fomin 1963]:

$$\frac{\delta\mathcal{F}}{\delta\mathbf{h}}(\mathbf{x}) = \mathbf{0} \qquad \text{for all } \mathbf{x} \in \partial\Omega. \tag{22}$$

## 3.3 Discretization of Boundary Data

*Parameterization.* To enable numerical optimization, we represent the unknown boundary function $\mathbf{h}(\mathbf{x})$ by a vector of $M$ parameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_M)$. For a general set of possibly higher-order basis functions $\{\phi_m\}_{m=1}^M$ defined along the boundary $\partial\Omega$, we write




$$\mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}) = \sum_{m=1}^{M} \theta_m\,\phi_m(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \partial\Omega. \tag{23}$$

For a piecewise linear discretization of $\mathbf{h}$ along the boundary, $\phi_m(\mathbf{x}) : \partial\Omega \to \mathbb{R}$ is the linear interpolation weight (shown on the right), and $\theta_m \in \mathbb{R}^n$ is the vector-valued unknown at discrete boundary points. We write $\mathbf{u}_{\boldsymbol{\theta}} := \mathbf{u}_{\mathbf{h}_{\boldsymbol{\theta}}}$ for the corresponding biharmonic solution.

*Energy.* Substituting (23) into (19) yields the energy $E(\boldsymbol{\theta})$:

$$E(\boldsymbol{\theta}) = \mathcal{F}[\mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x})] = \frac{1}{2}\int_{\partial\Omega} \left\|\partial_{\mathbf{n}}\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}) - \mathbf{g}(\mathbf{x})\right\|^2 \mathrm{d}S(\mathbf{x}). \tag{24}$$

Differentiating Eq. (24) with respect to $\theta_m$ yields the partials:

$$\frac{\partial E}{\partial \theta_m} = \int_{\partial\Omega} \partial_{\theta_m}\left(\partial_{\mathbf{n}}\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x})\right)^{\mathrm{T}} \cdot \left(\partial_{\mathbf{n}}\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}) - \mathbf{g}(\mathbf{x})\right) \mathrm{d}S(\mathbf{x}). \tag{25}$$

*Monte Carlo Approximation.* To evaluate the boundary integrals in Eq. (24) and Eq. (25) without discretization, we approximate the boundary integrals using $K_{\partial\Omega}$ Monte Carlo samples $\mathbf{x}_i \sim p(\mathbf{x})$ on $\partial\Omega$ with corresponding weights $w_i = \frac{1}{K_{\partial\Omega} \cdot p(\mathbf{x}_i)}$. The energy becomes:

$$E(\boldsymbol{\theta}) \approx \frac{1}{2} \sum_{i=1}^{K_{\partial\Omega}} w_i \left\| \partial_{\mathbf{n}}\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_i) - \mathbf{g}(\mathbf{x}_i) \right\|^2. \tag{26}$$

With total arclength $L = \int_{\partial\Omega} \mathrm{d}S(\mathbf{x})$, the samples $\mathbf{x}_i$ are uniformly distributed with the probability $p(\mathbf{x}_i) = 1/L$. Differentiating Eq. (26), the partial with respect to a parameter $\theta_m$ is estimated as

$$\frac{\partial E}{\partial \theta_m} \approx \sum_{i=1}^{K_{\partial\Omega}} w_i \cdot \partial_{\theta_m}\left(\partial_{\mathbf{n}}\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_i)\right)^{\mathrm{T}} \cdot \left(\partial_{\mathbf{n}}\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_i) - \mathbf{g}(\mathbf{x}_i)\right). \tag{27}$$

The gradient calculation thereby reduces to evaluating $\partial_{\mathbf{n}}\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x})$ and its parameter derivative $\partial_{\theta_m}\left(\partial_{\mathbf{n}}\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_i)\right)$ at sampled boundary points.

### 3.4 Normal Derivative Estimation

Several options are available to estimate normal derivatives. For spatial derivatives in the interior, the mean value theorem can be used [Sawhney and Crane 2020], which requires a non-zero ball radius around the evaluation point, making the method ill-conditioned near the domain boundary. Miller et al. [2023] sampled an offset point $(\mathbf{x}+h\mathbf{n}_{\mathbf{x}})$ from the boundary, and combined WoS samples from a sphere around the offset point. Using boundary value caching, they addressed the amplifying noise of long walks in Neumann dominated scenes. Miller et al. [2024a] have shown that a backward finite difference, which was previously used to estimate boundary solutions for bi-directional walks [Qi et al. 2022], has similar bias but less variance than the offset-sphere estimator. Yu et al. [2024] developed an off-centered estimator for normal derivatives at Dirichlet boundaries, which uses an off-centered antithetic sampling on the largest possible sphere, placed along the normal direction. Yu et al. [2025] extended the method to Neumann boundaries and avoided the hyper-singular kernels of Miller et al. [2023, 2024a] by moving the derivative off the kernel using integration by parts, improving the estimation in Neumann dominated scenes and on Neumann boundaries. Apart from normal derivatives, other work was concerned with differentiation with respect to boundary conditions [Miller et al. 2024a; Yilmazer et al. 2024], and with PDE-constrained shape optimization [Henrot and Pierre 2018], considering Reynolds transport theorem to differentiate with respect to shape parameters [Miller et al. 2024a].

*Finite-Difference.* Since we work on scenes with Dirichlet conditions only, we follow Miller et al. [2024a] and estimate the normal derivative with a backward finite difference using a two-point stencil, which offers first-order accuracy in step size $h$:

$$\partial_{\mathbf{n}}\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}) \approx \frac{1}{h}\left(\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}+h\mathbf{n}_{\mathbf{x}}) - \mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x})\right). \tag{28}$$

The value at $\mathbf{x} \in \partial\Omega$ is given by the Dirichlet condition, i.e., $\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}) = \mathbf{f}(\mathbf{x})$. We require $h > \epsilon$ such that the other point $(\mathbf{x}+h\mathbf{n}_{\mathbf{x}})$ is outside

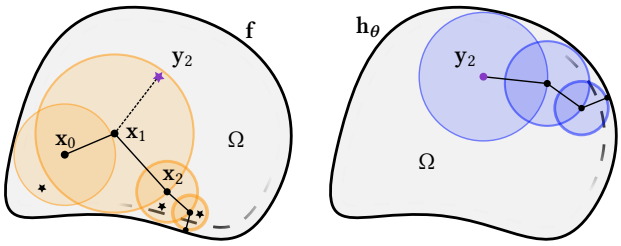



Fig. 3. Overview of our nested WoS estimators. For a current guess of the unknown $\mathbf{h}_{\boldsymbol{\theta}}$, we solve the *outer* Poisson problem (left) in Eqs. (35)–(36), which requires source samples (shown as star), here for example at $\mathbf{y}_2$. There, a nested WoS solves the *inner* Laplace problem (right) in Eqs. (31)–(32). This strategy is applied to outer walks originating in the domain Ω (for rendering) and close to the boundary (for finite differences during optimization).

of the $\epsilon$-shell. For higher-order stencils, correlated samples could be used, which is common in gradient domain rendering [Kettunen et al. 2015]. For differentiating Eq. (28) with respect to the unknown parameters $\theta_m$, we swap the differentiation order:

$$\partial_{\theta_m}\left(\partial_{\mathbf{n}}\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x})\right) = \partial_{\mathbf{n}}\left(\frac{\partial \mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x})}{\partial \theta_m}\right). \tag{29}$$

Similar to Eq. (28), we evaluate the normal derivative on the right-hand side of Eq. (29) with a two-point finite-difference stencil:

$$\partial_{\mathbf{n}}\left(\frac{\partial \mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x})}{\partial \theta_m}\right) \approx \frac{1}{h}\left(\frac{\partial \mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}+h\mathbf{n}_{\mathbf{x}})}{\partial \theta_m} - \mathbf{0}\right). \tag{30}$$

The last term has no contribution, since $\mathbf{f}(\mathbf{x})$ is independent of $\theta_m$, i.e., $\frac{\partial \mathbf{f}(\mathbf{x})}{\partial \theta_m} = \mathbf{0}$. In the following, we explain how $\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x})$ and $\frac{\partial \mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x})}{\partial \theta_m}$ are estimated numerically with Monte Carlo sampling.

### 3.5 Differentiable Monte Carlo Evaluation

For given boundary data $\mathbf{h}_{\boldsymbol{\theta}}$, we evaluate the two Dirichlet problems with WoS in a nested manner, as shown in Fig. 3. The inner WoS evaluates a Laplace equation for $\mathbf{v}_{\boldsymbol{\theta}}$, while the outer WoS evaluates a Poisson equation for $\mathbf{u}_{\boldsymbol{\theta}}$. In the following, we describe the estimators and calculate derivatives with respect to the boundary data $\mathbf{h}_{\boldsymbol{\theta}}$, effectively giving explicit formulas for path replay [Yilmazer et al. 2024]. Over the course of the optimization, $\mathbf{h}_{\boldsymbol{\theta}}$ will adjust to also match the Neumann condition of the Poisson problem.

*Differentiation of Laplace Equation.* The auxiliary function $\mathbf{v}_{\boldsymbol{\theta}}$ is defined as solution to a Laplace equation with Dirichlet condition:

$$\Delta \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}) = \mathbf{0} \qquad \text{for } \mathbf{x} \in \Omega, \tag{31}$$

$$\text{Dirichlet:} \quad \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}) = \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \partial\Omega. \tag{32}$$

We estimate the solution using WoS. Since there is no source term, expanding the recursive WoS walk from Eq. (7), starting from $\mathbf{x}_0$ and terminating after $N_{\mathrm{v}}$ steps, gives the expanded estimator:

$$\widehat{\mathbf{v}}_{\boldsymbol{\theta}}(\mathbf{x}_0) = \underbrace{\prod_{k=0}^{N_{\mathrm{v}}-1} \frac{\mathcal{P}^{\mathcal{B}}(\mathbf{x}_k, \mathbf{x}_{k+1})}{p^{\partial\mathcal{B}}(\mathbf{x}_{k+1})}}_{=1} \cdot \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_{N_{\mathrm{v}}}) = \mathbf{h}_{\boldsymbol{\theta}}(\mathbf{x}_{N_{\mathrm{v}}}). \tag{33}$$

If the samples are drawn *uniformly* with probability $p^{\partial\mathcal{B}}(\mathbf{x}_{k+1}) = \frac{1}{2\pi R}$, then the entire product cancels (=1). Inserting the discretization

of the unknown $\mathbf{h}_\theta$ from Eq. (23) into Eq. (33) and differentiating with respect to a particular degree of freedom $\theta_m$ we get:

$$\frac{\partial \widehat{\mathbf{v}}_\theta(\mathbf{x}_0)}{\partial \theta_m} = \mathbf{I}_{n\times n} \cdot \phi_m(\mathbf{x}_{N_\mathrm{v}}), \tag{34}$$

where $\mathbf{I}_{n\times n}$ is the $n \times n$ identity matrix, and $\mathbf{x}_{N_\mathrm{v}}$ is the reached end point on the Dirichlet boundary.

*Differentiation of Poisson Equation.* Similarly, we evaluate the Poisson equation with only its Dirichlet boundary condition:

$$\Delta \mathbf{u}_\theta(\mathbf{x}) = \widehat{\mathbf{v}}_\theta(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \Omega, \tag{35}$$

$$\text{Dirichlet:} \qquad \mathbf{u}_\theta(\mathbf{x}) = \mathbf{f}(\mathbf{x}) \qquad \text{for } \mathbf{x} \in \partial\Omega, \tag{36}$$

using Eq. (33) as source term $\widehat{\mathbf{v}}_\theta(\mathbf{x})$.
Expanding the recursive WoS estimate from Eq. (7) with source terms, we get:

$$\begin{aligned}\widehat{\mathbf{u}}_\theta(\mathbf{x}_0) = &\underbrace{\prod_{k=0}^{N_\mathbf{u}-1} \frac{\mathcal{P}^{\mathcal{B}}(\mathbf{x}_k, \mathbf{x}_{k+1})}{p^{\partial\mathcal{B}}(\mathbf{x}_{k+1})}}_{=1} \cdot \mathbf{f}(\mathbf{x}_{N_\mathbf{u}}) \\ &+ \sum_{k=0}^{N_\mathbf{u}-1} \underbrace{\left( \prod_{j=0}^{k-1} \frac{\mathcal{P}^{\mathcal{B}}(\mathbf{x}_j, \mathbf{x}_{j+1})}{p^{\partial\mathcal{B}}(\mathbf{x}_{j+1})} \right)}_{=1} \frac{\mathcal{G}^{\mathcal{B}}(\mathbf{x}_k, \mathbf{y}_{k+1})}{p^{\mathcal{B}}(\mathbf{y}_{k+1})} \cdot \widehat{\mathbf{v}}_\theta(\mathbf{y}_{k+1})\end{aligned} \tag{37}$$

$$= \mathbf{f}(\mathbf{x}_{N_\mathbf{u}}) + \sum_{k=0}^{N_\mathbf{u}-1} \frac{\mathcal{G}^{\mathcal{B}}(\mathbf{x}_k, \mathbf{y}_{k+1})}{p^{\mathcal{B}}(\mathbf{y}_{k+1})} \cdot \widehat{\mathbf{v}}_\theta(\mathbf{y}_{k+1}). \tag{38}$$

We again use a uniform sampling on $\partial\mathcal{B}$, which is why the product vanishes. The source samples are importance-sampled proportional to the Green's function, i.e., $p^{\mathcal{B}}(\mathbf{y}_{k+1}) \propto \mathcal{G}^{\mathcal{B}}(\mathbf{x}_k, \mathbf{y}_{k+1})$. With Eq. (38), there is a Dirichlet contribution when reaching the end point $\mathbf{x}_{N_\mathbf{u}}$, and a contribution from each source sample $\mathbf{y}_{k+1}$ along the way. Differentiating with respect to a parameter $\theta_m$ gives:

$$\frac{\partial \widehat{\mathbf{u}}_\theta(\mathbf{x}_0)}{\partial \theta_m} = \sum_{k=0}^{N_\mathbf{u}-1} \frac{\mathcal{G}^{\mathcal{B}}(\mathbf{x}_k, \mathbf{y}_{k+1})}{p^{\mathcal{B}}(\mathbf{y}_{k+1})} \cdot \frac{\partial \widehat{\mathbf{v}}_\theta(\mathbf{y}_{k+1})}{\partial \theta_m}. \tag{39}$$

We use Eq. (34) as estimator for $\frac{\partial \widehat{\mathbf{v}}_\theta(\mathbf{y}_{k+1})}{\partial \theta_m}$. Note that $m$ is implicit, i.e., $m$ depends on the boundary that is reached in the Laplace problem.

## 3.6 Acceleration of Convergence

*Reservoir Sampling.* Evaluating Eq. (39) requires a summation over all $N_\mathbf{u}$ source samples, which is prohibitively expensive, due to the nested Monte Carlo walks in the Laplace problem. We employ reservoir sampling for selecting one sample uniformly from a stream of unknown length [Vitter 1985]. We consider source events:

$$\mathbf{y}_{k+1} \in \mathcal{B}(\mathbf{x}_k), \qquad W_{k+1} = \frac{\mathcal{G}^{\mathcal{B}}(\mathbf{x}_k, \mathbf{y}_{k+1})}{p^{\mathcal{B}}(\mathbf{y}_{k+1})}. \tag{40}$$

Applied to the sequence of source events $(W_1, \mathbf{y}_1), \ldots, (W_{N_\mathbf{u}}, \mathbf{y}_{N_\mathbf{u}})$ generated by a WoS walk, the method proceeds as follows:

(1) Initialize a counter $k = 0$ and an empty reservoir.
(2) Each time a new source event $(W_{k+1}, \mathbf{y}_{k+1})$ is generated:
   (a) Increment $k \leftarrow k + 1$.
   (b) With probability $1/k$ place the source event in the reservoir.

After the walk, the reservoir contains exactly one of the $N_\mathbf{u}$ events, and the probability that the $k$-th event was stored is $\frac{1}{N_\mathbf{u}}$. Lastly, we incorporate the selection probability in both Eq. (38) and (39), and insert the chosen reservoir, e.g., $(W_{k+1}, \mathbf{y}_{k+1})$, from Eq. (40):

$$\widehat{\mathbf{u}}_\theta(\mathbf{x}_0) = \mathbf{f}(\mathbf{x}_{N_\mathbf{u}}) + N_\mathbf{u} \cdot W_{k+1} \cdot \widehat{\mathbf{v}}_\theta(\mathbf{y}_{k+1}), \tag{41}$$

$$\frac{\partial \widehat{\mathbf{u}}_\theta(\mathbf{x}_0)}{\partial \theta_m} = N_\mathbf{u} \cdot W_{k+1} \cdot \frac{\partial \widehat{\mathbf{v}}_\theta(\mathbf{y}_{k+1})}{\partial \theta_m}. \tag{42}$$

Vitter [1985] also discussed the selection of multiple source events from a stream, which we compare with later. We recommend selecting one event to retain comparable workload across GPU threads. Reservoir sampling is needed for good GPU utilization, but comes at the cost of increased variance, which we address next.

*Mean Value Caching.* Instead of estimating $\widehat{\mathbf{v}}_\theta$ from only a single WoS source sample $\mathbf{y}_{k+1}$, we employ mean value caching [Bakbouk and Peers 2023] to reuse walks in the Laplace problem. Each iteration, we *uniformly distribute* new cache samples $\mathbf{x}_p$ in the domain $\Omega$, estimate $\widehat{\mathbf{v}}_\theta(\mathbf{x}_p)$ using Eq. (33), and store $m$ and $\phi_m(\mathbf{x}_{N_\mathrm{v}})$ for later evaluation of the partial in Eq. (34). To evaluate a source sample $\mathbf{y}_{k+1}$ in the Poisson problem, we utilize the mean value theorem and average the contributions of all $N_p$ precomputed samples that are inside the largest ball $\mathcal{B}(\mathbf{y}_{k+1})$, i.e., $\forall \mathbf{x}_p \in \mathcal{B}(\mathbf{y}_{k+1})$:

$$\widehat{\mathbf{v}}_\theta(\mathbf{x}) = \frac{1}{|\mathcal{B}(\mathbf{x})|} \int_{\mathcal{B}(\mathbf{x})} \widehat{\mathbf{v}}_\theta(\mathbf{y})\, \mathrm{d}\mathbf{y} \approx \frac{1}{|N_p|} \sum_{\mathbf{x}_p \in \mathcal{B}(\mathbf{x})} \underbrace{\widehat{\mathbf{v}}_\theta(\mathbf{x}_p)}_{\text{Eq. (33)}}. \tag{43}$$

The partials in Eq. (34) are distributed atomically using the stored $m$ and $\phi_m(\mathbf{x}_{N_\mathrm{v}})$ per visited cache sample $\mathbf{x}_p$. With changing boundary values $\mathbf{h}_\theta$, the cache is recomputed once per iteration.

## 3.7 Unbiased Multi-Sampled Energy and Gradient

The product of two Monte Carlo estimators introduces a bias in form of covariance, i.e., $\mathrm{E}(XY) = \mathrm{E}(X)\mathrm{E}(Y) + \mathrm{Cov}(X, Y)$. Thus, for an unbiased estimate of the energy in Eq. (26) and its gradient in Eq. (27), we compute $\partial_\mathbf{n} \widehat{\mathbf{u}}_\theta(\mathbf{x}_i)$ and $\partial_{\theta_m}\big(\partial_\mathbf{n} \widehat{\mathbf{u}}_\theta(\mathbf{x}_i)\big)$ from the same random number sequence, while an independent $\partial_\mathbf{n} \overline{\mathbf{u}}_\theta(\mathbf{x}_i)$ is computed from an uncorrelated random number sequence [Miller et al. 2024a]:

$$E(\boldsymbol{\theta}) \approx \frac{1}{2} \sum_{i=1}^{K_{\partial\Omega}} w_i \left(\partial_\mathbf{n} \widehat{\mathbf{u}}_\theta(\mathbf{x}_i) - \mathbf{g}(\mathbf{x}_i)\right)^{\mathrm{T}} \left(\partial_\mathbf{n} \overline{\mathbf{u}}_\theta(\mathbf{x}_i) - \mathbf{g}(\mathbf{x}_i)\right), \tag{44}$$

$$\frac{\partial E}{\partial \theta_m} \approx \sum_{i=1}^{K_{\partial\Omega}} w_i \cdot \partial_{\theta_m}\big(\partial_\mathbf{n} \widehat{\mathbf{u}}_\theta(\mathbf{x}_i)\big)^{\mathrm{T}} \cdot \left(\partial_\mathbf{n} \overline{\mathbf{u}}_\theta(\mathbf{x}_i) - \mathbf{g}(\mathbf{x}_i)\right). \tag{45}$$

Thus, in practice, the term $\left(\partial_\mathbf{n} \overline{\mathbf{u}}_\theta(\mathbf{x}_i) - \mathbf{g}(\mathbf{x}_i)\right)$ is first estimated with a multi-sample Monte Carlo estimator. Subsequently, the walks used to evaluate $\partial_{\theta_m}\big(\partial_\mathbf{n} \widehat{\mathbf{u}}_\theta(\mathbf{x}_i)\big)$ determine the discrete boundary element $m$ that is hit, and the corresponding partial derivative is accumulated for that parameter. Accumulating these partials over multiple iterations before performing a gradient descent step yields a multi-sample estimator of the energy gradient.

## 3.8 Gradient-Based Optimization of Boundary Data

The Monte Carlo estimator in Eq. (45) provides an unbiased estimate of the gradient of the discrete energy $E(\boldsymbol{\theta})$. We update the

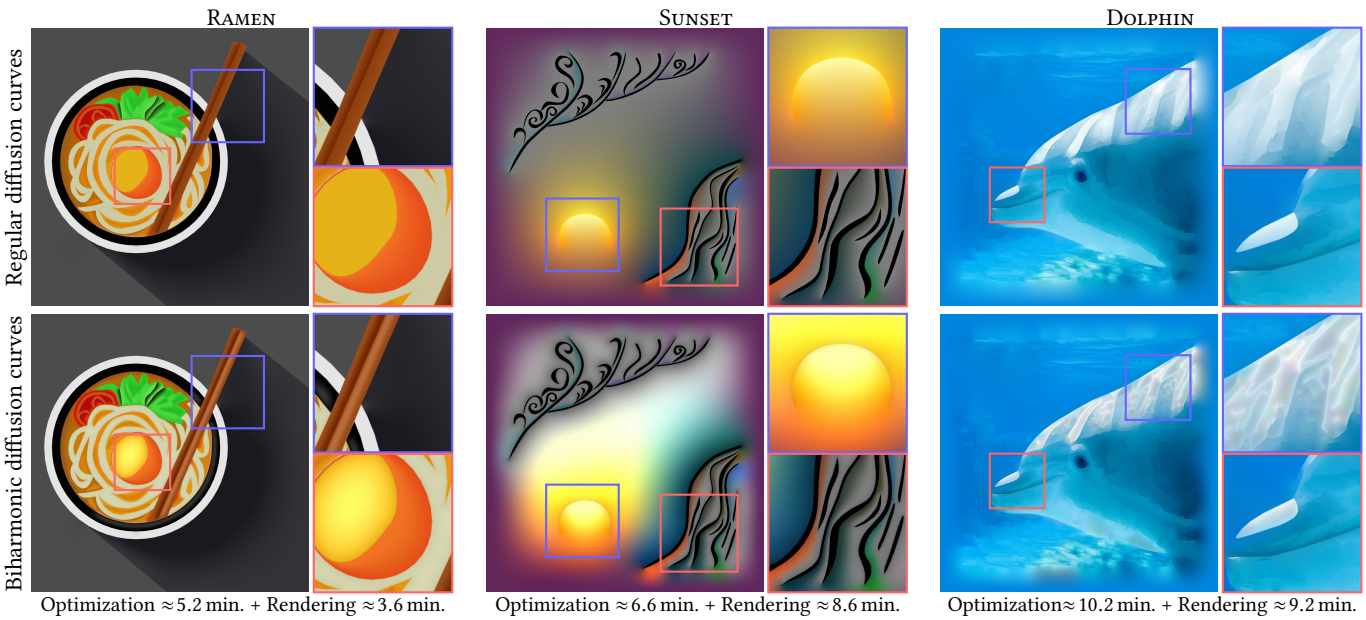



Fig. 4. Qualitative comparison of regular diffusion curves (top row) and biharmonic diffusion curves (bottom row) rendered at a resolution of 1024 × 1024 pixels with 8,192 walks per pixel, after optimization for 4,096 iterations. The regular diffusion curves were rendered as Laplace problem with our input Dirichlet data, while the biharmonic diffusion curves were rendered with our inverse Monte Carlo method (bottom row). This figure illustrates the input Dirichlet data to our method and gives examples of complex color gradients, obtainable with the biharmonic formulation.

parameters $\boldsymbol{\theta}$ using gradient descent with step size $\alpha > 0$:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \alpha \nabla_{\boldsymbol{\theta}} E\big(\boldsymbol{\theta}^{(t)}\big), \quad \nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) = \left(\frac{\partial E}{\partial \theta_1}, \ldots, \frac{\partial E}{\partial \theta_M}\right)^{\top}, \quad (46)$$

and begin with the initial guess $\boldsymbol{\theta}^{(0)} = \mathbf{0}$. The optimization loop is outlined in Alg. 1. Since the gradient estimator in Eq. (45) is computed from Monte Carlo samples, it naturally introduces stochastic noise, which is well suited for stochastic gradient methods such as Adam [Kingma and Ba 2017]. The optimization iteratively improves the boundary function $\mathbf{h}_{\boldsymbol{\theta}}$ until the estimated Neumann mismatch $\partial_{\mathbf{n}}\mathbf{u}_{\boldsymbol{\theta}} - \mathbf{g}$ becomes negligible. This yields a numerical approximation of the optimal boundary data $\mathbf{h}^*$ defined in Eq. (20). Reference implementations in C++ for both 1D and 2D cases, as well as our test scenes, are available on GitHub[1].

**Algorithm 1** Optimization Loop for Biharmonic Diffusion Curves

Initialize parameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_M)$
**while** not converged **do**
  $(\mathbf{x}_i, \mathbf{n}_i) \leftarrow$ sample boundary point $\mathbf{x}_i \sim p(\mathbf{x})$
  $w_i = \frac{1}{K_{\partial\Omega}\, p(\mathbf{x}_i)}$ , $\overline{\mathbf{u}}_{\boldsymbol{\theta}}(\mathbf{x}_i + h\mathbf{n}_i) \leftarrow \mathbf{0}$ , $\frac{\partial E}{\partial \theta_m} \leftarrow 0$
  *// Estimate normal derivative* $\partial_{\mathbf{n}}\overline{\mathbf{u}}_{\boldsymbol{\theta}}$
  **for** $k = 1$ to $K_{\overline{\mathbf{u}}}$ **do**
    $\overline{\mathbf{u}}_{\boldsymbol{\theta}}(\mathbf{x}_i + h\mathbf{n}_i) \leftarrow \overline{\mathbf{u}}_{\boldsymbol{\theta}}(\mathbf{x}_i + h\mathbf{n}_i) + \frac{1}{K_{\overline{\mathbf{u}}}}\mathrm{WoS}(\mathbf{x}_i + h\mathbf{n}_i)$ ▸ Eq. (41)
  $\partial_{\mathbf{n}}\overline{\mathbf{u}}_{\boldsymbol{\theta}}(\mathbf{x}_i) \approx \frac{1}{h}\big(\overline{\mathbf{u}}_{\boldsymbol{\theta}}(\mathbf{x}_i + h\mathbf{n}_i) - \mathbf{f}(\mathbf{x}_i)\big)$ ▸ Eq. (28)
  *// Estimate* $\theta$*-derivatives*
  **for** $k = 1$ to $K_{\widehat{\mathbf{u}}}$ **do**
    $\left(\frac{\partial \widehat{\mathbf{u}}_{\boldsymbol{\theta}}(\mathbf{x}_i + h\mathbf{n}_i)}{\partial \theta_m},\ m\right) \leftarrow \partial\mathrm{WoS}(\mathbf{x}_i + h\mathbf{n}_i)$ ▸ Eq. (42)
    $\partial_{\theta_m}(\partial_{\mathbf{n}}\mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_i)) \approx \frac{1}{h}\left(\frac{\partial \mathbf{u}_{\boldsymbol{\theta}}(\mathbf{x}_i + h\mathbf{n}_i)}{\partial \theta_m} - \mathbf{0}\right)$ ▸ Eqs. (29)–(30)
    $g_m = w_i \cdot \partial_{\theta_m}\big(\partial_{\mathbf{n}}\widehat{\mathbf{u}}_{\boldsymbol{\theta}}(\mathbf{x}_i)\big)^{\mathrm{T}} \cdot \big(\partial_{\mathbf{n}}\overline{\mathbf{u}}_{\boldsymbol{\theta}}(\mathbf{x}_i) - \mathbf{g}(\mathbf{x}_i)\big)$ ▸ Eq. (45)
    $\frac{\partial E}{\partial \theta_m} \leftarrow \frac{\partial E}{\partial \theta_m} + g_m$
  *// Update gradient*
  **for** $m = 1$ to $M$ **do**
    $\theta_m \leftarrow \theta_m - \alpha \frac{\partial E}{\partial \theta_m}$ ▸ Eq. (46)

*Convergence.* Both $E$ and $\frac{\partial E}{\partial \theta_m}$ are estimated numerically with Monte Carlo integration. Monte Carlo gradients lead under mild step size conditions to the unique minimizer of strictly convex functionals if the gradients are unbiased and the variance is bounded. While noise is helpful in early iterations, it is necessary to lower the noise in later iterations to ensure convergence. Thus, we follow Miller et al. [2024a] and empirically set both of the sample counts $K_{\widehat{\mathbf{u}}}^{(i)}$ and $K_{\overline{\mathbf{u}}}^{(i)}$ in iteration $(i)$ with an annealing scheduler $K_*^{(i)} = K_*^0 + \exp\left(\log\left(\frac{K_*^I}{K_*^0}\right) \cdot \frac{i}{I}\right)$. Thereby, $K_*^0$ is the initial and $K_*^I$ is the final sample count in iteration $I$. For all scenes, we set $K_{\widehat{\mathbf{u}}}^0 = K_{\overline{\mathbf{u}}}^0 = 1$, $K_{\widehat{\mathbf{u}}}^I = 4$, and $K_{\overline{\mathbf{u}}}^I = 16$, which was sufficient in practice.

## 4 Results

*Qualitative Results.* We begin with a side-by-side view of Monte Carlo renderings of regular diffusion curves and biharmonic diffusion curves in Fig. 4. The top row illustrates the Dirichlet data $\mathbf{f}$ that is given as input to our method, and the bottom row gives examples of complex color gradients obtained with the biharmonic formulation. Close-ups highlight selected non-linear color gradients. In this figure, as well as in all other results shown throughout the paper, the images were rendered at a resolution of 1024 × 1024 pixels.

[1]Available at https://github.com/Paul-Hi/biharmonic-monte-carlo.

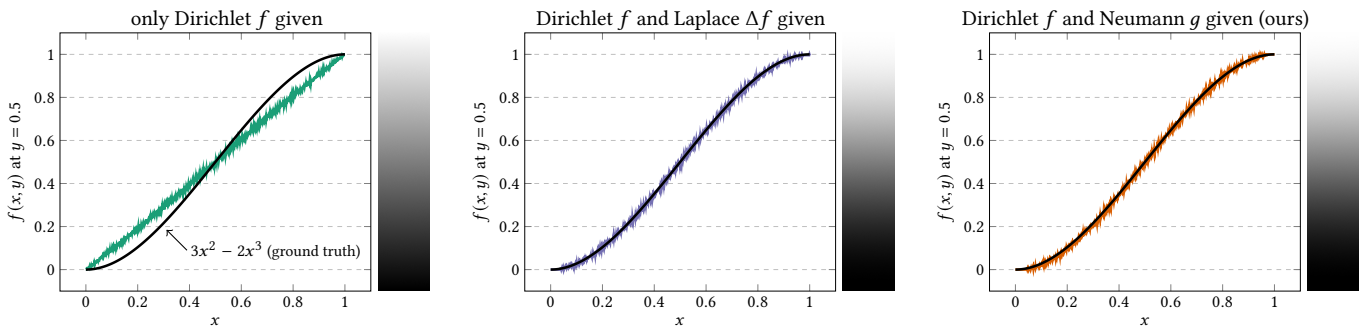



Fig. 5. To confirm correctness, we reconstruct the analytic biharmonic *smoothstep* function. The Laplace problem (left) receives Dirichlet data only and incorrectly reconstructs a straight line. With Dirichlet and Laplace given, the biharmonic solution (middle) can be estimated exactly with a nested Walk-on-Spheres [Sawhney and Crane 2020]. If Dirichlet and normal derivative are given instead (right), our inverse Monte Carlo method is needed, which reconstructs the analytic ground truth (shown as black curve) exactly. Here estimated with 8,192 samples for the harmonic solution (left) and 16,384 samples for the biharmonic solutions (middle and right). The vertical color gradients depict the reconstructed intensity profile, which is identical for the biharmonic solutions.

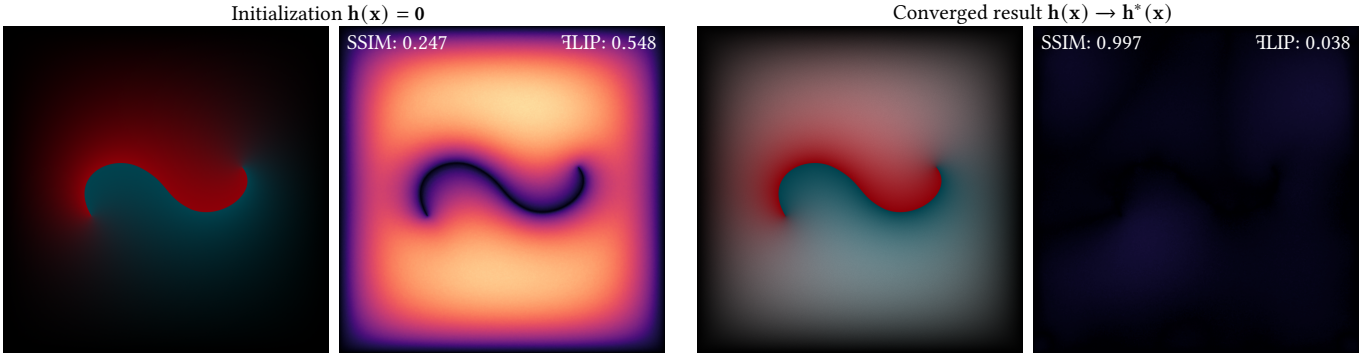



Fig. 6. First, we construct a ground truth, setting $\mathbf{f}$ and $\mathbf{h}^*$ on all curve boundaries. Here, we set $\mathbf{h}^* = \mathbf{4}$. Having estimated the normal derivative $\widehat{\mathbf{g}} := \partial_{\mathbf{n}}\mathbf{u}$ numerically, we check whether our optimization, given only $\mathbf{f}$ and $\widehat{\mathbf{g}}$, can restore the Laplacian $\mathbf{h}^*$. Here, the resulting image and the ꟻLIP image are shown.

*Correctness.* To confirm correctness, we reconstruct an analytic biharmonic function from Dirichlet and Neumann boundary data. A simple one-dimensional biharmonic function is the *smoothstep* function $f(x) = 3x^2 - 2x^3$. We consider it in the 2D domain $[0, 1]^2$ as $f(x, y) := f(x)$, which results in a smooth color gradient. A section of the color gradient is visualized in Fig. 5 along with an intensity plot. The Laplace equation (left) interpolates only the Dirichlet boundary data, resulting in an incorrect straight profile at $y = 1/2$. If we directly solve the nested Walk-on-Spheres (middle) with Dirichlet $f$ and Laplace $\Delta f$ data given on the boundary, similar to Sawhney and Crane [2020], we can recover the cubic interpolant perfectly. For biharmonic diffusion curves (right), we are interested in problems with Dirichlet data $f$ and Neumann data $g$ instead, which our inverse Monte Carlo method reconstructs perfectly, as well, showing that we are able to reconstruct the analytic ground truth. In Fig. 6, we test convergence on a scene with double-sided interior boundaries, for which we first construct a ground truth by providing Dirichlet data $\mathbf{f}$ and a known Laplace $\mathbf{h}^*$ on all boundaries $\partial\Omega$. In this boundary condition setting, the nested Walk-on-Spheres [Sawhney and Crane 2020] is applicable, from which we numerically estimate the normal derivative $\widehat{\mathbf{g}} := \partial_{\mathbf{n}}\mathbf{u}$. When phrasing the problem inversely, i.e., when imposing the Dirichlet $\mathbf{f}$ and the estimated Neumann data $\widehat{\mathbf{g}}$, we expect to reconstruct the same image using our inverse Monte Carlo solver. We report the perceptual error metrics *structural similarity index measure* (SSIM, ↑) and ꟻLIP (↓), which indicate good convergence, here shown after 6 *min* (4,224 iterations).

*Runtime Measurements.* We measured the runtime of our CUDA GPU implementation on a workstation equipped with an AMD Ryzen 9 7950X CPU and an NVIDIA RTX 5090 GPU. Fig. 7 shows that the runtime per iteration scales nearly linearly when increasing the number of boundary samples $K_{\partial\Omega}$, reservoir bins, and the number of samples for the two uncorrelated estimators for $\overline{\mathbf{u}}$ and $\widehat{\mathbf{u}}$. All measurements have been averaged over 128 runs. As expected, the trend is linear with and without mean value caching, where the exact slope depends on the cache size. Because of the linearly increasing computational cost, we recommend a reservoir size of 1, i.e., only one source sample is selected each iteration. This lower number of source samples is compensated by mean value caching.

*Convergence.* We plot convergence sequences for Ramen and Zephyr in Fig. 8, reporting a convergence plot of the estimated energy $E(\boldsymbol{\theta})$ in Eq. (44) on the left, as well as result images and ꟻLIP images after a fixed number of iterations on the right. The initial frame starts with $\mathbf{h}_{\boldsymbol{\theta}} = \mathbf{0}$, showing the largest differences in areas

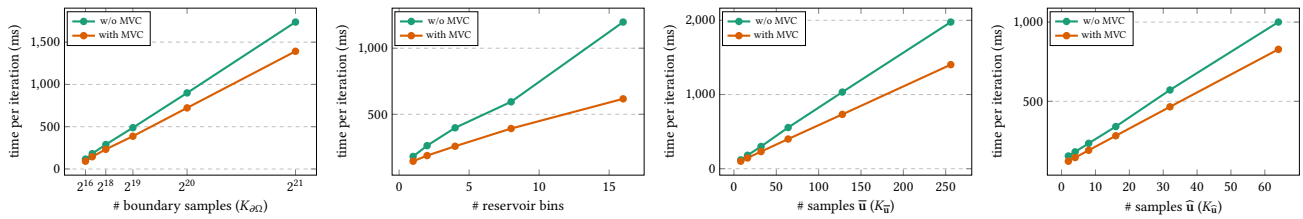



Fig. 7. The runtime increases nearly linearly when increasing the number of boundary samples ($K_{\partial\Omega}$), reservoir bins, $\overline{\mathbf{u}}$ samples ($K_{\overline{\mathbf{u}}}$), and $\widehat{\mathbf{u}}$ samples ($K_{\widehat{\mathbf{u}}}$). The trend is visible both with (orange) and without (green) mean value caching enabled. The slope depends on the cache size. Averaged over 128 runs.

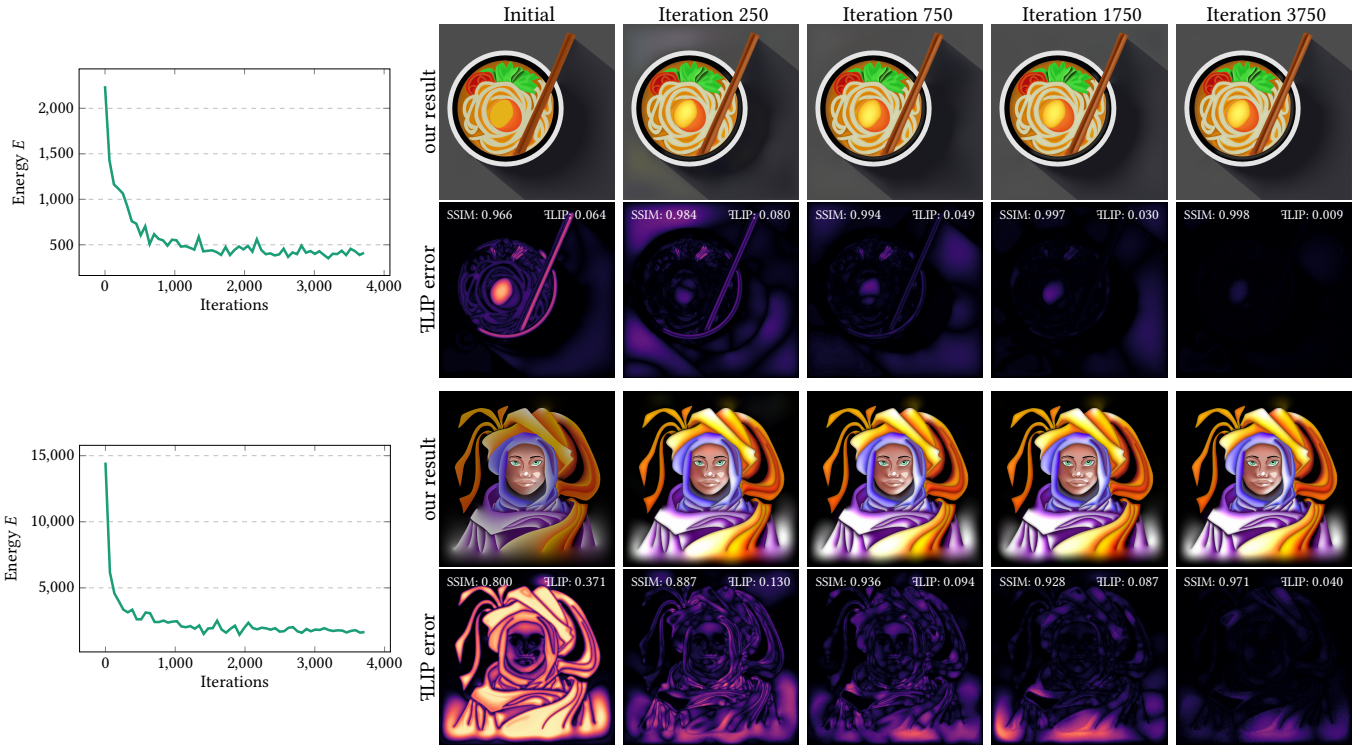



Fig. 8. We show the convergence on complex vector graphics scenes by plotting the change in energy $E$ over 3,750 iterations and measuring the visual deviation from a fully converged optimization result. We employ the perceptual error metrics *structural similarity index measure* (SSIM, ↑) and ꟻLIP (↓). The ꟻLIP error map is shown below each image, and the mean ꟻLIP error is reported in the corner. The converged reference was optimized with over 4,000 iterations. Despite the inevitable noise in the stochastic estimation of the energy $E$, a convergence trend is clearly visible. Likewise, the image sequence and the perceptual error map follow this trend, yielding a better approximation of the reference solution with increasing iteration count.

where the Neumann data $\mathbf{g}$ has largest influence. Apparent in both the perceptual metrics SSIM and ꟻLIP, as well as in the ꟻLIP image itself, the error decreases during the stochastic gradient descent.

*Equal-Time Comparison.* To assess the utility of mean value caching in our implementation, we ran the optimization in Fig. 9 without (top) and with (bottom) mean value caching and report results for the Ramen scene after equal time of optimization. The energy residual (left) decays slightly faster with mean value caching, with the speed-up depending on the sparseness of the scene. Visually, the difference in the resulting images is more pronounced, which is visible in the ꟻLIP images, and in the SSIM and ꟻLIP metric. Notably mean value caching helps most in regions far from boundary geometry, due to the larger cache query radius.

*Wall Clock Time and Early Results.* To obtain a ground truth for the comparisons in Figs. 1, 10 (top), and 4, we optimized with 4,096 iterations and rendered with 8,192 walks per pixel, which took in minutes (optimization/rendering): Zephyr (≈ 6.9/≈ 5.4), Ramen (≈ 5.2/≈ 3.6), Sunset (≈ 6.6/≈ 8.6), and Dolphin (≈ 10.2/≈ 9.2). Our gradient-based optimization for solving a fourth-order PDE requires approximately twice as many walks per pixel to reach the noise level of regular diffusion curves without source term. In fact, the regular diffusion curve results, rendered with 4,096 walks

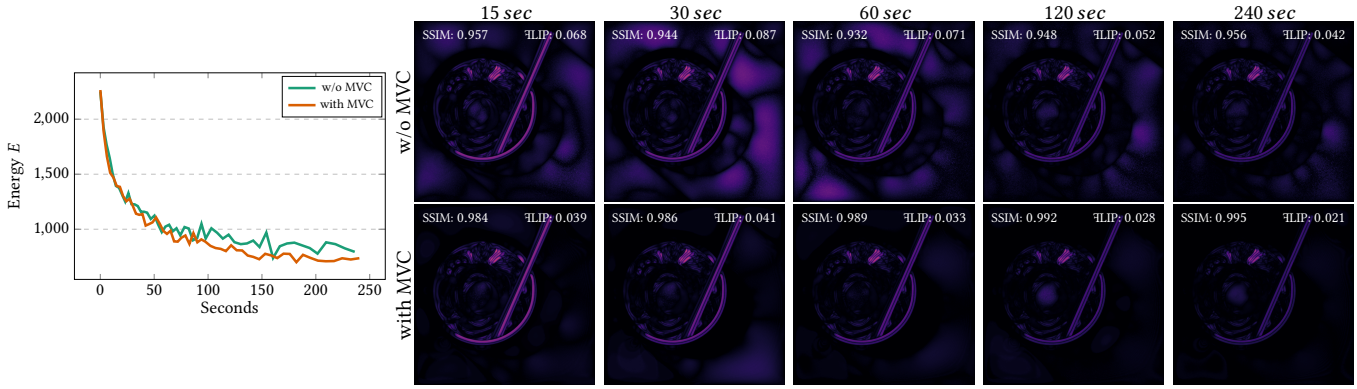



Fig. 9. Equal-time comparison of our method without (top) and with (bottom) mean value caching (MVC). The energy estimate $E$ (left) reduces slightly faster with MVC during the stochastic gradient descent and the overall convergence behaves smoother. The FLIP error image and the perceptual error metrics indicate faster and more robust convergence with MVC. Due to the stochastic nature of the descent, structures (such as in the bottom right) may appear briefly, but vanishes after sufficient time.


Fig. 10. Comparison of fully converged solutions (top) with early results, optimized for 5 seconds and rendered for 5 seconds (bottom).

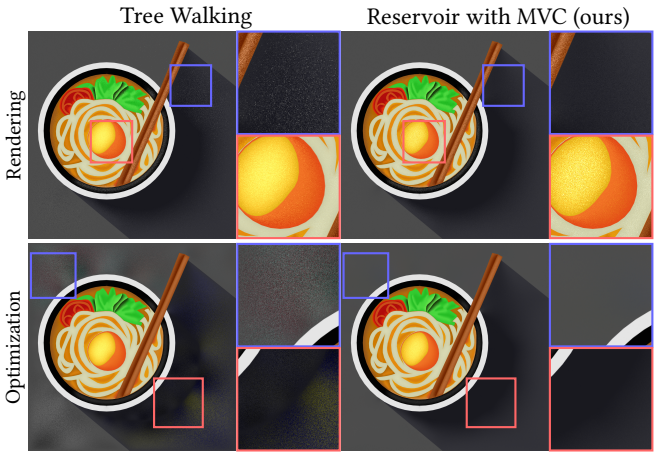



Fig. 11. Equal-time comparison of tree walking [Sawhney and Crane 2020] (left) and our method (right). Top: rendering result after one second. Bottom: result after 10 seconds of optimization and 5 seconds of rendering.

per pixel, took ≈2.3 (Zephyr), ≈1.5 (Ramen), ≈3.6 (Sunset), and ≈ 4 (Dolphin) minutes, respectively. In practice, a significantly smaller amount of iterations/walks is needed. In Fig. 10 (bottom), we present early results after just 10 seconds for all scenes (5 seconds optimized, 5 seconds rendered), demonstrating high quality after a short time.

*Tree Walking.* Sawhney and Crane [2020] proposed nested walks for solving biharmonic equations with Dirichlet and Laplacian boundary data. Their tree walking method can be integrated into our approach in place of using reservoir sampling and mean value caching. Rather than sampling the inner walk from the reservoir, each source sample $\mathbf{y}_{k+1}$ is connected to the next point on the outer walk $\mathbf{x}_{k+1}$. Weighting the endpoint of the outer walk $\overline{\mathbf{x}}_N$ by the off-centered Poisson kernel $\widehat{\mathbf{v}}_{\boldsymbol{\theta}}(\mathbf{y}_{k+1}) = \frac{\mathcal{P}^{\mathcal{B}}(\mathbf{y}_{k+1},\mathbf{x}_{k+1})\mathbf{h}_{\boldsymbol{\theta}}(\overline{\mathbf{x}}_N)}{p^{\partial\mathcal{B}}(\mathbf{x}_{k+1})}$ directly estimates the result of the walk. Fig. 11 (top) compares both strategies after equal time. While tree walking converges faster in regions with small area and uniform boundary data, our approach is advantageous in large areas away from boundaries. Due to the correlated paths, tree walking exhibits higher variance, which negatively impacts the convergence of the gradient-based solver, see Fig. 11 (bottom). In contrast, our approach becomes more stable due to the reduced variance, which benefits fast previews as well.

## 5 Conclusion

Biharmonic diffusion curves offer direct control over colors and color gradients along curve boundaries, which requires higher-order smoothness in the interior in form of a biharmonic equation. Enforcing Dirichlet conditions (for color) and Neumann conditions (for color gradient) is theoretically sufficient for a unique solution [Ilbery et al. 2013], however, solving for the function $\mathbf{u}(\mathbf{x})$ is difficult, since in the split formulation the Poisson equation is over-constrained, while the coupled Laplace equation has unknown boundary data. Thus, the rendering becomes an inverse problem, in which the unknown boundary data has to be recovered. To this end, we developed a variational energy minimization that utilizes Walk-on-Spheres as it allows for point-wise estimation of normal derivatives along the

boundary. Our work is at the intersection of recent and advanced topics in Monte Carlo PDEs, including normal derivative estimation, inverse rendering via path replay, as well as reservoir sampling and mean value caching for acceleration and variance reduction.

Our method requires a discretization of the unknown boundary data $\mathbf{h}$. We discretized the boundary geometry into piecewise-linear segments and found this discretization to be sufficient for our scenes. Future work could investigate adaptive representations and higher-order basis functions. Biharmonic diffusion curves set Dirichlet constraints on $\mathbf{u}$ and Neumann constraints on $\partial\mathbf{u}_\mathbf{n}$. We absorbed the latter via a Dirichlet constraint in the inner Laplace problem of the split formulation, which led to pure Dirichlet problems. Constraining the Laplace $\Delta\mathbf{u}$ instead of $\partial\mathbf{u}_\mathbf{n}$ is trivial, as it is equivalent to a Dirichlet constraint. However, when combining $\mathbf{u}$ with $\partial\Delta\mathbf{u}_\mathbf{n}$, or $\partial\mathbf{u}_\mathbf{n}$ with $\Delta\mathbf{u}$ or $\partial\Delta\mathbf{u}_\mathbf{n}$, or when absorbing the $\partial\mathbf{u}_\mathbf{n}$ constraint through a Neumann condition in the Laplace problem, the problem is no longer pure Dirichlet and would likely benefit from a Walk-on-Stars [Sawhney et al. 2023; Yu et al. 2025]. We used a two-point stencil for the finite-difference estimation of normal derivatives. In the future, we would like to experiment with the off-centered estimator of Yu et al. [2024]. Alternatively, higher-order finite-difference stencils could be used. Here, the walks of the additional stencil points could be correlated, similar to the gradient estimators used in gradient domain rendering [Kettunen et al. 2015]. Reconnecting correlated paths through suitable shift mappings, for example where one sample is seen as an off-centered sample of the other [Bao et al. 2025; Himmler and Günther 2025], could be an interesting step for the future to ensure that paths end on the same boundary segment. To eliminate the inherent bias that is introduced by finite differences, debiasing could be used [Misso et al. 2022]. Incorporating recent strategies for variance reduction, such as harmonic caching [Zhou et al. 2025], could further improve stability and convergence.

## Acknowledgments

The Zephyr and Dolphin scene originate from Orzan et al. [2008]. The Sunset is from Tian and Günther [2025]. This work was supported by DFG grant no. GU 945/3-1.

## References

Ghada Bakbouk and Pieter Peers. 2023. Mean Value Caching for Walk on Spheres. In *Eurographics Symposium on Rendering* (Delft, The Netherlands). The Eurographics Association, 1–10. doi:10.2312/sr.20231120

Seungbae Bang, Kirill Serkh, Oded Stein, and Alec Jacobson. 2023. An Adaptive Fast-Multipole-Accelerated Hybrid Boundary Integral Equation Method for Accurate Diffusion Curves. *ACM Transactions on Graphics (TOG)* 42, 6, Article 215 (dec 2023), 28 pages. doi:10.1145/3618374

Anchang Bao, Jie Xu, Enya Shen, and Jianmin Wang. 2025. Off-Centered WoS-Type Solvers with Statistical Weighting. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers (SA Conference Papers '25)*. Association for Computing Machinery, New York, NY, USA, Article 127, 11 pages. doi:10.1145/3757377.3763852

John C. Bowers, Jonathan Leahey, and Rui Wang. 2011. A Ray Tracing Approach to Diffusion Curves. *Computer Graphics Forum* 30, 4 (2011), 1345–1352. doi:10.1111/j.1467-8659.2011.01994.x

Simon Boyé, Pascal Barla, and Gaël Guennebaud. 2012. A vectorial solver for free-form vector gradients. *ACM Trans. Graph.* 31, 6, Article 173 (Nov. 2012), 9 pages. doi:10.1145/2366145.2366192

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. doi: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. doi:10.1145/3731196

Martin Costabel. 1987. Principles of boundary element methods. *Computer Physics Reports* 6, 1-6 (1987), 243–274. doi:10.1016/0167-7977(87)90014-1

Michael Czekanski, Benjamin Faber, Margaret Fairborn, Adelle Wright, and David Bindel. 2024. Walking on Spheres and Talking to Neighbors: Variance Reduction for Laplace's Equation. arXiv:2404.17692 [physics.comp-ph]

B. Fraeijs De Veubeke. 1974. Variational principles and the patch test. *Internat. J. Numer. Methods Engrg.* 8, 4 (1974), 783–801. doi:10.1002/nme.1620080408

Janick Martinez Esturo, Maik Schulze, Christian Rössl, and Holger Theisel. 2013. Poisson-based tools for flow visualization. In *IEEE Pacific Visualization Symposium (PacificVis)*. IEEE, Sydney, NSW, Australia, 241–248. doi:10.1109/PacificVis.2013.6596151

Mark Finch, John Snyder, and Hugues Hoppe. 2011. Freeform vector graphics with controlled thin-plate splines. *ACM Trans. Graph.* 30, 6 (Dec. 2011), 1–10. doi:10.1145/2070781.2024200

Izrail Moiseevitch Gelfand and S. V. Fomin. 1963. *Calculus of variations*. Prentice-Hall Inc., Englewood Cliffs, N. J. doi:10.2307/3611748

Antoine Henrot and Michel Pierre. 2018. *Shape variation and optimization*. Vol. 28. European Mathematical Society Publishing House, Zuerich, Switzerland. doi:10.4171/178

Paul Himmler and Tobias Günther. 2025. Conformal First Passage for Epsilon-free Walk-on-Spheres. *ACM Trans. Graph.* 44, 4, Article 40 (July 2025), 11 pages. doi:10.1145/3730942

Fei Hou, Qian Sun, Zheng Fang, Yong-Jin Liu, Shi-Min Hu, Hong Qin, Aimin Hao, and Ying He. 2018. Poisson vector graphics (PVG). *IEEE Transactions on Visualization and Computer Graphics* 26, 2 (2018), 1361–1371. doi:10.1109/TVCG.2018.2867478

Tianyu Huang, Jingwang Ling, Shuang Zhao, and Feng Xu. 2025. Guiding-Based Importance Sampling for Walk on Stars. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers (SIGGRAPH Conference Papers '25)*. Association for Computing Machinery, New York, NY, USA, Article 3, 12 pages. doi:10.1145/3721238.3730593

Chi-Ok Hwang, Sungpyo Hong, and Jinwoo Kim. 2015. Off-centered "Walk-on-Spheres" (WOS) algorithm. *J. Comput. Phys.* 303 (2015), 331–335. doi:10.1016/j.jcp.2015.10.002

Peter Ilbery, Luke Kendall, Cyril Concolato, and Michael McCosker. 2013. Biharmonic diffusion curve images from boundary elements. *ACM Trans. Graph.* 32, 6, Article 219 (Nov. 2013), 12 pages. doi:10.1145/2508363.2508426

Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. 2011. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.* 30, 4, Article 78 (July 2011), 8 pages. doi:10.1145/2010324.1964973

Stefan Jeschke, David Cline, and Peter Wonka. 2009. A GPU Laplacian Solver for Diffusion Curves and Poisson Image Editing. *ACM Transactions on Graphics* 28, 5 (2009), 1–8. doi:10.1145/1618452.1618462

Markus Kettunen, Marco Manzi, Miika Aittala, Jaakko Lehtinen, Frédo Durand, and Matthias Zwicker. 2015. Gradient-domain path tracing. *ACM Trans. Graph.* 34, 4, Article 123 (July 2015), 13 pages. doi:10.1145/2766997

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs.LG]

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* (Sydney, NSW, Australia). ACM, New York, NY, USA, Article 34, 10 pages. doi:10.1145/3610548.3618141

Shibo Liu, Tielin Dai, Ligang Liu, and Xiao-Ming Fu. 2025. Polynomial 2D Biharmonic Coordinates for High-order Cages. *ACM Trans. Graph.* 44, 4, Article 77 (July 2025), 10 pages. doi:10.1145/3730887

Shufang Lu, Wei Jiang, Xuefeng Ding, Craig S Kaplan, Xiaogang Jin, Fei Gao, and Jiazhou Chen. 2019. Depth-aware image vectorization and editing. *The Visual Computer* 35, 6 (2019), 1027–1039. doi:10.1007/s00371-019-01671-0

James McCann and Nancy S. Pollard. 2008. Real-time gradient-domain painting. *ACM Trans. Graph.* 27, 3 (Aug. 2008), 1–7. doi:10.1145/1360612.1360692

Bailey Miller, Rohan Sawhney, Keenan Crane, and Ioannis Gkioulekas. 2023. Boundary Value Caching for Walk on Spheres. *ACM Trans. Graph.* 42, 4, Article 82 (jul 2023), 11 pages. doi: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. doi: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, Article 41 (jul 2024), 18 pages. doi:10.1145/3658153

Zackary Misso, Benedikt Bitterli, Iliyan Georgiev, and Wojciech Jarosz. 2022. Unbiased and consistent rendering using biased estimators. *ACM Trans. Graph.* 41, 4, Article 48 (July 2022), 13 pages. doi:10.1145/3528223.3530160

Mervin E Muller. 1956. Some continuous Monte Carlo methods for the Dirichlet problem. *The Annals of Mathematical Statistics* 27, 3 (1956), 569–589. doi:10.1214/aoms/1177728169

Alexandrina Orzan, Adrien Bousseau, Holger Winnemöller, Pascal Barla, Joëlle Thollot, and David Salesin. 2008. Diffusion curves: a vector representation for smooth-shaded images. *ACM Trans. Graph.* 27, 3 (Aug. 2008), 1–8. doi:10.1145/1360612.1360691

Wai-Man Pang, Jing Qin, Michael Cohen, Pheng-Ann Heng, and Kup-Sze Choi. 2011. Fast rendering of diffusion curves with triangles. *IEEE Computer Graphics and Applications* 32, 4 (2011), 68–78. doi:10.1109/MCG.2011.86

Romain Prévost, Wojciech Jarosz, and Olga Sorkine-Hornung. 2015. A Vectorial Framework for Ray Traced Diffusion Curves. *Computer Graphics Forum* 34, 1 (2015), 253–264. doi:10.1111/cgf.12510

Yang Qi, Dario Seyb, Benedikt Bitterli, and Wojciech Jarosz. 2022. A bidirectional formulation for Walk on Spheres. *Computer Graphics Forum (Proc. Eurographics Symposium on Rendering)* 41, 4 (2022), 51–62. doi:10.1111/cgf.14586

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. doi:10.1145/3550454.3555450

Karl Sabelfeld and Elena Shkarupa. 2003. Functional Random Walk on Spheres algorithm for biharmonic equation: optimization and error estimation. *Monte Carlo Methods and Applications* 9, 1 (2003), 51–65. doi:10.1515/156939603322587461

Karl Karlovich Sabelfeld. 1982. Vector algorithms in the Monte-Carlo method for solving systems of second-order elliptic equations and Lamés equation. *Doklady Akademii Nauk* 262, 5 (1982), 1076–1080. https://www.mathnet.ru/eng/dan45069 (in Russian).

Karl K. Sabelfeld. 1991. *Monte Carlo Methods in Boundary Value Problems* (1 ed.). Springer, Berlin, Heidelberg.

Karl K. Sabelfeld and Nikolai A. Simonov. 1994. *Random Walks on Boundary for Solving PDEs*. De Gruyter, Berlin, Boston. doi:10.1515/9783110942026

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, Article 123 (Aug. 2020), 18 pages. doi:10.1145/3386569.3392374

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.* 42, 4, Article 80 (jul 2023), 20 pages. doi:10.1145/3592398

Rohan Sawhney, Dario Seyb, Wojciech Jarosz, and Keenan Crane. 2022. Grid-free Monte Carlo for PDEs with spatially varying coefficients. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)* 41, 4 (July 2022). doi:10/gqjn65

Reinhard Scholz. 1978. A mixed method for 4th order problems using linear finite elements. *RAIRO. Analyse numérique* 12, 1 (1978), 85–90. https://www.numdam.org/item/M2AN_1978__12_1_85_0/

Ryusuke Sugimoto, Christopher Batty, and Toshiya Hachisuka. 2024. 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. doi: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. doi:10.1145/3592109

Timothy Sun, Papoj Thamjaroenporn, and Changxi Zheng. 2014. Fast multipole representation of diffusion curves and points. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 53–1. doi:10.1145/2601097.2601187

Xin Sun, Guofu Xie, Yue Dong, Stephen Lin, Weiwei Xu, Wencheng Wang, Xin Tong, and Baining Guo. 2012. Diffusion curve textures for resolution independent texture mapping. *ACM Trans. Graph.* 31, 4, Article 74 (July 2012), 9 pages. doi:10.1145/2185520.2185570

Jean-Marc Thiery, Élie Michel, and Jiong Chen. 2024. Biharmonic Coordinates and their Derivatives for Triangular 3D Cages. *ACM Trans. Graph.* 43, 4, Article 138 (July 2024), 17 pages. doi:10.1145/3658208

Xingze Tian and Tobias Günther. 2025. Monte Carlo Methods for 2D Flow Visualization. In *Eurographics Conference on Visualization Short Papers*. Eurographics, Luxembourg City, Luxembourg. doi:10.2312/evs.20251089

Xingze Tian and Tobias Günther. 2024. A Survey of Smooth Vector Graphics: Recent Advances in Representation, Creation, Rasterization, and Image Vectorization. *IEEE Transactions on Visualization and Computer Graphics* 30, 3 (2024), 1652–1671. doi:10.1109/TVCG.2022.3220575

Xingze Tian and Tobias Günther. 2025. Unified Smooth Vector Graphics: Modeling Gradient Meshes and Curve-based Approaches Jointly as Poisson Problem. *IEEE Transactions on Visualization and Computer Graphics* 31, 10 (2025), 7848–7861. doi:10.1109/TVCG.2025.3558263

Jeffrey S. Vitter. 1985. Random sampling with a reservoir. *ACM Trans. Math. Softw.* 11, 1 (March 1985), 37–57. doi:10.1145/3147.3165

Ofir Weber, Roi Poranne, and Craig Gotsman. 2012. Biharmonic Coordinates. *Computer Graphics Forum* 31, 8 (2012), 2409–2422. doi:10.1111/j.1467-8659.2012.03130.x

Ekrem Fatih Yilmazer, Delio Vicini, and Wenzel Jakob. 2024. Solving Inverse PDE Problems using Grid-Free Monte Carlo Estimators. *ACM Trans. Graph.* 43, 6, Article 175 (Nov. 2024), 18 pages. doi:10.1145/3687990

Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, and Heung-Yeung Shum. 2004. Mesh editing with poisson-based gradient field manipulation. In *ACM SIGGRAPH 2004 Papers* (Los Angeles, California) *(SIGGRAPH '04)*. Association for Computing Machinery, New York, NY, USA, 644–651. doi:10.1145/1186562.1015774

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 254 (Dec. 2025), 16 pages. doi:10.1145/3763333

Zihan Yu, Lifan Wu, Zhiqian Zhou, and Shuang Zhao. 2024. A Differential Monte Carlo Solver For the Poisson Equation. In *ACM SIGGRAPH 2024 Conference Papers* (Denver, CO, USA) *(SIGGRAPH '24)*. Association for Computing Machinery, New York, NY, USA, Article 11, 10 pages. doi:10.1145/3641519.3657460

Shuang Zhao, Frédo Durand, and Changxi Zheng. 2017. Inverse diffusion curves using shape optimization. *IEEE Transactions on Visualization and Computer Graphics* 24, 7 (2017), 2153–2166. doi:10.1109/TVCG.2017.2721400

Zihong Zhou, Eugene d'Eon, Rohan Sawhney, and Wojciech Jarosz. 2025. Harmonic Caching for Walk on Spheres. *ACM Trans. Graph.* 44, 6, Article 253 (Dec. 2025), 15 pages. doi:10.1145/3763322