Monte Carlo Steklov Operators for Large-Scale Geometry Processing in the Wild

ARMAN MAESUMI*, Brown University, USA
TANISH MAKADIA*, Brown University, USA
ARUNA ANDERSON†, Loyola Marymount University, USA
ORAS PHONGPANANGAM†, Brown University, USA
JUSTIN SOLOMON, Massachusetts Institute of Technology, USA
DANIEL RITCHIE, Brown University, USA

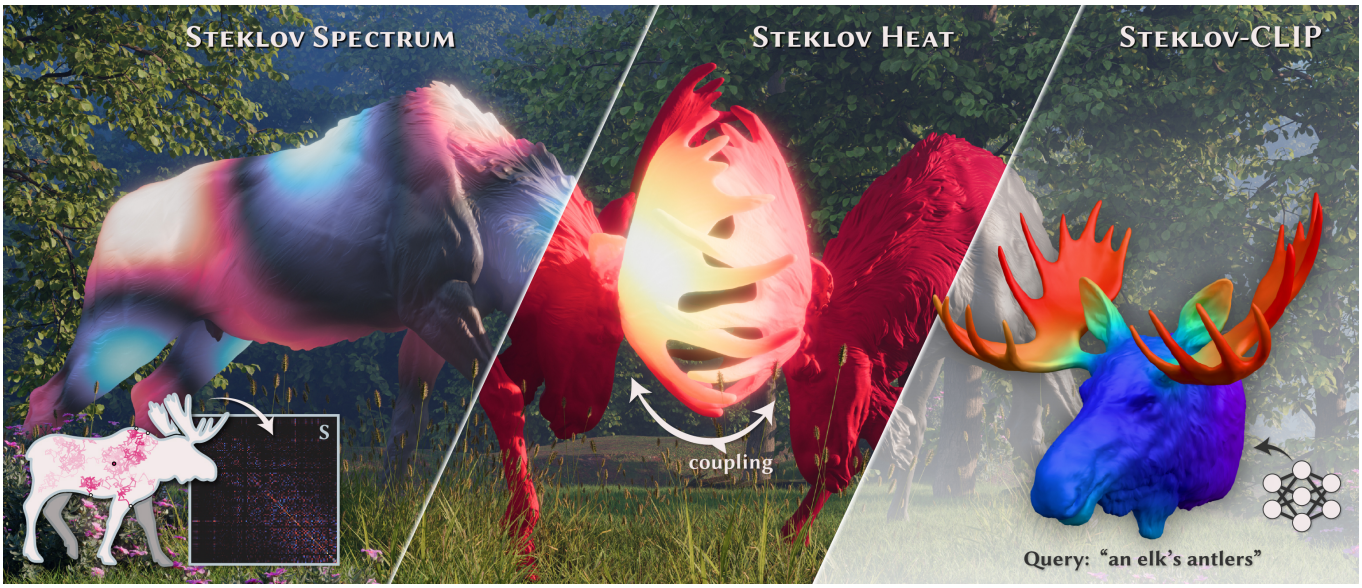

Fig. 1. We present a Monte Carlo method for estimating the Dirichlet-to-Neumann (DtN) operator and its associated Steklov eigenmodes, enabling fast and robust volumetric analysis of *in-the-wild* shapes. *Left:* We formulate a Brownian process that resolves entries of a discrete DtN operator—the 87th eigenvector of this operator is visualized on the moose, which contains 2 million faces. *Center:* Our method applies to both *interior* and *exterior* operators—the latter encodes coupling behavior between disconnected geometries. *Right:* Using our estimated Steklov operators, we train a mesh-based CLIP model that learns meaningful global and dense shape representations, which is able to isolate semantic components via text, e.g. "`an elk's antlers.`"

Intrinsic methods fill the default toolbox for geometry processing on meshes. Intrinsic operators, in particular the Laplacian, underlie methods that require invariance to isometry and have hence been employed in many algorithms for shape analysis, learning, and editing. However, intrinsic methods are predicated on assumptions that quickly become brittle when working with in-the-wild geometry, where (i) mesh quality is not guaranteed, and (ii) many meshes are modeled with multiple connected components. In such settings, volumetric constructions are better-defined, since restrictions on surface topology can be relaxed. This paper presents a Monte Carlo method for estimating the Dirichlet-to-Neumann (DtN) operator—a boundary-to-boundary volumetric operator—and its associated Steklov eigenmodes. We build on recent developments in Monte Carlo geometry processing by casting this boundary operator itself as the subject of estimation. The DtN operator, defined through a volumetric stochastic process, is then generalized to the exterior domain, where it couples disconnected components through the surrounding ambient space. We show that our method is orders of magnitude faster than existing boundary-element approaches for computing Steklov spectra while remaining robust to poor triangulations, high-resolution meshes, and multi-component geometry. To demonstrate this scalability, we compute interior and exterior Steklov eigenspectra for approximately 450,000 shapes from the uncurated Objaverse dataset. We incorporate these operators into Steklov-CLIP, a mesh-based neural network that uses volumetric spectral operators for large-scale contrastive 3D representation learning. The resulting network learns semantically meaningful global and dense shape representations, illustrating that geometrically-principled volumetric operators can be made practical at the scale of modern 3D datasets.

Additional Key Words and Phrases: Spectral shape analysis, Monte Carlo methods, representation learning

1 Introduction

The intrinsic Laplacian operator has long served as the workhorse of geometry processing. Because it depends only on the metric of a

*These authors contributed equally to this work.
†These authors contributed equally to this work as co-second authors.

surface, it is invariant to isometric deformation, admits sparse local discretizations, and underpins methods for shape editing [Sorkine et al. 2007; Jacobson et al. 2011], parametrization [Desbrun et al. 2002; Sawhney and Crane 2017; Lévy et al. 2023], correspondence [Ovsjanikov et al. 2012; Litany et al. 2017], decomposition [Huang et al. 2009; Reuter 2010], and geometric deep learning [Smirnov and Solomon 2021; Sharp et al. 2022; Wiersma et al. 2022; Maesumi et al. 2025].

The assumptions behind many intrinsic pipelines, however, have become restrictive in the age of contemporary geometry processing, which typically features large, minimally-curated shape collections whose meshes vary substantially in quality [Deitke et al. 2023]. The discretized operators (e.g., the cotangent Laplacian matrix) employed by intrinsic methods often expect clean, single-component, manifold meshes with reasonable element quality, and these operators are highly sensitive to spurious changes in topology that are incidental to how the mesh was modeled.

For such data, an extrinsic or volumetric viewpoint can be more natural. The occupied volume of a shape is frequently a more stable geometric signal than the surface induced by scanning, reconstruction, 3D generative models, or manual authoring; moreover, volumetric operators capture structure that intrinsic methods necessarily ignore. Indeed, volumetric and extrinsic boundary-based methods have proven useful in a range of shape-analysis settings (see discussion in Section 2).

Among extrinsic operators, the *Dirichlet-to-Neumann (DtN) operator* is particularly appealing for its applications to geometry processing [Wang et al. 2018]. Given a scalar function on a closed surface, the DtN operator returns the normal flux of its (volumetric) harmonic extension through the boundary. Although it acts purely as a boundary-to-boundary operator, it encodes volumetric geometry: its eigenpairs are the *Steklov modes*, which provide an extrinsic spectral description of shape. Wang et al. [2018] demonstrated the practical utility of this operator for geometry processing, showing that Steklov spectra stably introduce extrinsic information to geometry processing pipelines while avoiding tetrahedralization via a boundary element (BEM) formulation.

Scalability and robustness are key obstacles for volumetric methods in general. Classical finite-element and boundary-element methods for volumetric operators are substantially more expensive than the sparse intrinsic pipelines that dominate surface processing. Even when tetrahedralization is avoided, discretizations of volumetric operators are typically dense, and hence computing Steklov eigenspectra can require on the order of tens of minutes on meshes that are modest by the standards of current graphics datasets. Methods that rely on tetrahedralization similarly are accompanied by exorbitant up-front preprocessing costs that may fail or not fully preserve the original surface mesh (see Figure 4 and Table 1 from Dodik et al. [2025]). Even when one accepts these drawbacks, these pipelines may not terminate successfully, e.g. due to sensitivity to poor element quality or quadrature. These bottlenecks preclude large-scale volumetric analysis, especially for datasets containing high-resolution meshes with poor element quality.

Meanwhile, Monte Carlo methods have emerged as an attractive approach for solving volumetric elliptic boundary value problems in graphics [Sawhney and Crane 2020; Sawhney et al. 2023]. These

Fig. 2. The heat equations governed by interior and exterior Steklov eigenspectra exhibit non-trivial dynamics. We evolve a Dirac source distribution localized to a single keycap, and see that the interior heat evolves through the hollowed area beneath each cap, leaking into adjacent ones. The exterior heat evolves with a wider profile through the ambient space surrounding the keys. We note that the Commodore 64 keyboard has 2662 connected components that are triangulated very poorly, yet our method's estimated Steklov eigenmodes nevertheless give high-quality results.

works extend the Walk-on-Spheres (WoS) method [Muller 1956], which estimates the solution of Laplace's equation by simulating Brownian motion and does not require a volumetric mesh or dense intermediate constructions. WoS is progressive, naturally parallelizable, and comparatively insensitive to poor element quality or local geometric degeneracies [Sawhney and Crane 2020]. Most of these Monte Carlo methods, however, estimate *solutions* to PDEs at selected query points. For shape analysis, the object of interest is different: we would like to estimate the *boundary operator* itself, enabling downstream geometry processing algorithms.

In this paper, we introduce a Monte Carlo method for estimating Dirichlet-to-Neumann operators, and in particular their associated Steklov eigenspectra. We first demonstrate that a naïve Monte Carlo estimator for these operators results in shortcomings that mirror those of previous volumetric methods: dense linear algebra and compromised structure due to poor sampling dynamics (e.g., high variance). As an alternative, we use the Beurling-Deny Formula [Beurling and Deny 1958] to rewrite DtN operators, uncovering a Monte Carlo estimator that produces a positive semidefinite matrix from any finite number of samples. We decompose the rewritten integral forms of these operators into analytic and estimated parts, reducing sampling variance. We further observe that Steklov eigenspectra are well-approximated in a compact functional basis on the surface, bypassing dense intermediate constructions. Our estimators are implemented in CUDA and yield a highly-practical

pipeline for using DtN operators even on meshes with millions of elements or poor surface discretizations.

Further, by applying the DtN operator (and our estimator) to *exterior* domains, we obtain an operator with an appealing property for geometry processing: unlike local surface-based operators, which act independently on disconnected components, the exterior DtN operator *couples* boundary regions that are topologically disconnected on the mesh but interact through the ambient space (see Figures 2 and 3). We demonstrate the utility of this property on shapes with disconnected components and cavities, where it enables information to be propagated across disjoint regions.

We summarize our contributions as follows:

- We derive Monte Carlo estimators for the interior and exterior Dirichlet-to-Neumann operators on triangle meshes and estimate their Steklov eigenspectra efficiently without requiring a tetrahedral mesh or large dense intermediate matrices.
- We demonstrate the scalability and robustness of our approach by computing interior and exterior Steklov eigenspectra across ∼450K shapes from Objaverse, far exceeding what is possible with existing spectral volumetric methods.
- We incorporate these operators into a neural network and perform mesh-based 3D representation learning via contrastive training. We demonstrate that our network, Steklov-CLIP, learns rich representations on meshes that are aligned with an existing text-image contrastive model (i.e. CLIP). The network is further fine-tuned to produce dense representations that facilitate spatial queries (e.g. zero-shot semantic selection of parts).

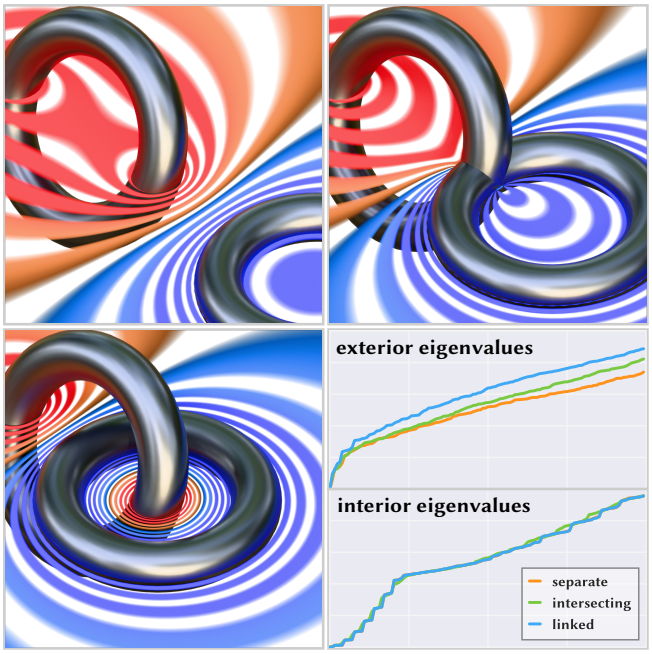

Fig. 3. The exterior DtN operator—being defined through the exterior harmonic extension (visualized as iso-potentials around each torus)—is sensitive to the relative positions and orientations of objects in a scene. By contrast, the interior DtN operator is relatively stable to such changes, due to the shapes being closed. This difference in behavior can be seen in their associated eigenspectra (bottom right).

2 Related Work

Extrinsic Geometry Processing. Geometry processing algorithms commonly build on the Laplace-Beltrami operator and its eigenspectrum due to its invariance to isometries. This invariance is also a limitation: a purely intrinsic operator cannot distinguish shapes that share the same metric—e.g. a human mesh in different poses. A broad line of work therefore replaces or augments the Laplacian with operators that expose extrinsic information; see the comprehensive survey of Wang and Solomon [2019] for further background. Representative examples include discrete Dirac operators [Liu et al. 2017; Ye et al. 2018] and functional shape-difference methods that separate intrinsic and extrinsic quantities [Corman et al. 2017]. Closest to our setting is Steklov geometry processing [Wang et al. 2018], which uses the Dirichlet-to-Neumann operator—in particular its eigenspectrum—as an extrinsic spectral operator. Wang et al. [2018] demonstrated that the Steklov eigenmodes can be inserted into standard spectral geometry pipelines, but they rely on a boundary element method that limits scalability to large meshes and is brittle in the face of poorly triangulated ones. Our work builds on this boundary-operator viewpoint, but instead estimates interior and exterior DtN operators by stochastic (Monte Carlo) sampling, which allows us to apply this machinery to large meshes with degenerate triangulations, and is orders of magnitude faster than existing volumetric spectral methods.

Spectral Geometry. Spectral shape analysis represents geometry through eigenvalues and eigenfunctions of geometric operators. Seminal applications include spectral shape descriptors built from the Laplace-Beltrami eigenmodes [Sun et al. 2009; Bronstein and Kokkinos 2010; Aubry et al. 2011]; functional maps, which represent dense correspondence as compact linear operators in reduced eigenbases [Ovsjanikov et al. 2012], as well as later learning-based functional map pipelines [Litany et al. 2017]; finally, select methods for geometric learning define feature transformations using filters defined in the spectral domain [Smirnov and Solomon 2021; Sharp et al. 2022; Gao et al. 2024]. The Steklov eigenspectrum is attractive because its eigenfunctions live on the surface, while the underlying operator is induced by a *volumetric* harmonic extension. The exterior variant of this operator further changes the information encoded by its associated spectrum since the relevant harmonic process takes place in the ambient complement of the volume. Through this process, the exterior Steklov spectrum couples disconnected geometry based on harmonic accessibility through the surrounding space. Components that are topologically disjoint may nevertheless interact strongly when Brownian motions through the exterior domain readily have access to them, while intervening "barriers" can attenuate this interaction, making the process—and hence the spectrum—highly informative of the geometry at hand.

Monte Carlo PDE. The stochastic foundations of our method go back to Kakutani's representation of harmonic functions through the lens of Brownian motion and Muller's Walk-on-Spheres algorithm for the Dirichlet problem [Kakutani 1944; Muller 1956]. Recent work in graphics has revived these ideas to motivate grid-free solvers for elliptic PDEs on complex geometry. In particular, Sawhney and

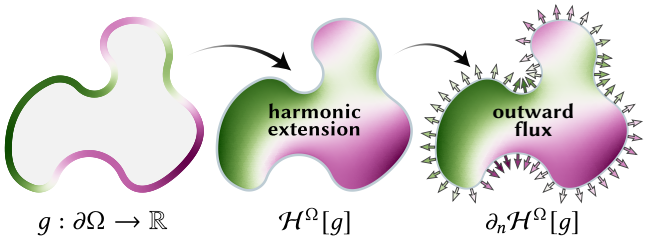

Fig. 4. The Dirichlet-to-Neumann operator maps scalar boundary functions, g, to the outward normal derivative of their harmonic extension.

Crane [2020] introduced Muller's Walk-on-Spheres method to the graphics community and extended it to include efficient estimation of solution derivatives and variance reduction techniques. Subsequent work expanded on this methodology to PDEs with spatially-varying coefficients [Sawhney et al. 2022], infinite domains via the Kelvin transform [Nabizadeh et al. 2021], mixed Neumann/Dirichlet and Robin boundary conditions [Sawhney et al. 2023; Miller et al. 2024b], and further variance reduction techniques have been considered [Miller et al. 2023; Li et al. 2024; Huang et al. 2025], along with methods for fast recomputation using local solution operators [Jambon et al. 2026]. Boundary-integral Monte Carlo methods provide another perspective on this stochastic framework by considering random walks directly on surfaces [Karlovič Sabel'fel'd and Simonov 1994; Sugimoto et al. 2023]. Differential Monte Carlo solvers estimate sensitivities or normal derivatives of PDE solutions, including derivatives with respect to boundary data or shape [Miller et al. 2024a; Yu et al. 2024]. The estimators introduced in this paper address a complementary goal—rather than estimating the solution, gradient, or derivative of a PDE at selected query points, we instead estimate *boundary operators* themselves and extract eigenspectra from them, to enable fast and robust volumetric spectral geometry processing.

3 Preliminaries

Our goal is to devise a Monte Carlo estimator for the Dirichlet-to-Neumann operator that enables rapid and robust approximation of its eigenspectrum. We begin with elementary definitions from PDEs that are necessary for defining this operator. We briefly introduce stochastic perspectives, which motivate Monte Carlo methods for such problems. Finally, we introduce the Dirichlet-to-Neumann operator, along with its corresponding Steklov eigenproblem.

Dirichlet Problem. Let $\Omega \subset \mathbb{R}^3$ be a bounded volumetric domain with smooth boundary $\partial\Omega$. Given a scalar-valued function $g : \partial\Omega \to \mathbb{R}$, the *Dirichlet Problem* seeks a harmonic function $u : \Omega \to \mathbb{R}$ on the interior that equals g on the surface:

$$\begin{cases} \Delta u = 0 & \text{in } \Omega, \\ u = g & \text{on } \partial\Omega, \end{cases} \tag{1}$$

where Δ is the Laplacian operator. The solution u uniquely extends g to the volume, and hence it is referred to as the *harmonic extension* of g. We denote the harmonic extension operator as $\mathcal{H}^\Omega[g] = u$.

Harmonic Measure and the Poisson Kernel. Let $x \in \Omega$ be a point in the domain's interior. The harmonic extension operator $\mathcal{H}^\Omega[g]$ defines a probability measure ω_x^Ω over the boundary by

$$u(x) = \int_{\partial\Omega} g(s)\, d\omega_x^\Omega(s). \tag{2}$$

Here, $\omega_x^\Omega(s)$ is the *harmonic measure* of a surface point $s \in \partial\Omega$ with respect to the interior point x. Since harmonic measure and surface area measure σ are both defined over the same measurable space $\partial\Omega$, the Radon-Nikodym Theorem gives a density $P_x^\Omega = d\omega_x^\Omega / d\sigma$, called the *Poisson kernel*. Thus, for any surface patch $A \subseteq \partial\Omega$,

$$\omega_x^\Omega(A) = \int_{s\in A} P_x^\Omega(s)\, d\sigma(s). \tag{3}$$

Hence, P_x^Ω is the *Radon-Nikodym derivative* of harmonic measure with respect to surface area. Combining this definition with Eq. 2, the Poisson kernel's utility comes from the fact that it solves the Dirichlet problem with a boundary integral

$$u(x) = \int_{\partial\Omega} g(s)\, P_x^\Omega(s)\, d\sigma(s). \tag{4}$$

A probabilistic interpretation of the Poisson kernel was given by Kakutani [1944], who showed that if B_t is a Brownian motion started at x and $\tau = \inf\{t \mid B_t \notin \Omega\}$ is the first time B_t exits the volume, then $\mathcal{H}^\Omega[g]$ is given by

$$u(x) = \mathbb{E}[g(B_\tau)], \tag{5}$$

proving that ω_x^Ω is a probability distribution over exit points B_τ with density P_x^Ω. To better represent this perspective, we denote the Poisson kernel as $P^\Omega(x \to s)$, giving the probability density that a Brownian motion originating from $x \in \Omega$ first hits the boundary at $s \in \partial\Omega$.

Connection to Green's Function. The Poisson kernel has an equivalent characterization through the *Green's function* of the Laplacian operator. The Green's function $G^\Omega(x, s)$ is defined by

$$\Delta_x G^\Omega(x, s) = \delta(x - s). \tag{6}$$

It describes the system's response at $x \in \Omega$ to a point-source placed at $s \in \Omega$. The Poisson kernel is recovered by taking the normal derivative of G^Ω at the boundary,

$$P^\Omega(x \to s) = \partial_{n_s} G^\Omega(x, s), \tag{7}$$

where n_s is the outward unit normal at $s \in \partial\Omega$.

Dirichlet-to-Neumann Operator. The Poisson integral (Eq. 4) maps boundary values g to their harmonic extension u. Similarly, the *Dirichlet-to-Neumann* (DtN) *operator*

$$\Lambda : H^{\frac{1}{2}}(\partial\Omega) \to H^{-\frac{1}{2}}(\partial\Omega) \quad \text{where} \quad g \mapsto \partial_n \mathcal{H}^\Omega[g] \tag{8}$$

maps g to the normal flux of u. Here, H^k denotes the Sobolev space of order k. This boundary-to-boundary operator can be viewed as a composition; first take the harmonic extension of Dirichlet data g to obtain u, and then take the outward normal derivative $\partial_n u$ to obtain Neumann data (as illustrated in Figure 4). Because Λ maps between different classes of boundary conditions, it falls into a broader family of operators known as *Poincaré-Steklov operators* [Agoshkov 1988].

Steklov Eigenspectrum. The *Steklov eigenproblem* seeks nonzero boundary data ψ whose harmonic extension has normal derivative proportional to its boundary trace. In other words, if $u = \mathcal{H}^\Omega[\psi]$ satisfies

$$\begin{cases} \Delta u = 0 & \text{in } \Omega, \\ \partial_n u = \lambda u & \text{on } \partial\Omega, \end{cases} \tag{9}$$

then $\Lambda\psi = \lambda\psi$. Thus ψ is an eigenfunction of Λ with eigenvalue λ. We refer to the corresponding eigenpairs (λ, ψ) as the *Steklov eigenmodes* of Λ. The primary interest of this paper lies in approximating this spectrum efficiently to enable downstream applications. We further discuss the utility of the Steklov eigenspectrum in Section 2.

Properties of DtN. The DtN operator has a natural bilinear form that reveals its core structure. For two boundary functions $f, g \in H^{\frac{1}{2}}(\partial\Omega)$, define

$$\mathcal{E}[f,g] := \int_{\partial\Omega} f(s)(\Lambda g)(s)\, ds. \tag{10}$$

Applying Green's first identity to the harmonic extensions $u_f = \mathcal{H}^\Omega[f]$ and $u_g = \mathcal{H}^\Omega[g]$ transforms this boundary integral into a volume integral,

$$\mathcal{E}[f,g] = \int_{\partial\Omega} f(\partial_n u_g)\, ds \tag{11}$$

$$= \int_\Omega \nabla u_f \cdot \nabla u_g\, dV. \tag{12}$$

The right-hand side is the *Dirichlet energy* inner product of two harmonic extensions. This identity has two immediate consequences.

(1) **Symmetry.** Since $\mathcal{E}[f,g] = \mathcal{E}[g,f]$, we know that Λ is a self-adjoint operator.
(2) **PSD.** Setting $f = g$ gives $\mathcal{E}[f,f] = \int_\Omega |\nabla u_f|^2\, dV \geq 0$. Hence, Λ is positive semidefinite.

The latter property in particular is central to the development of our core method in Section 4.

Walk-on-Spheres. The probabilistic identity in Eq. 5 suggests a method for estimating the harmonic extension of boundary data. By simulating a Brownian motion from x and recording where it first hits $\partial\Omega$, we can sample exit points $B_\tau \sim \omega_x^\Omega$ from a distribution matching the harmonic measure. Directly tracing a continuous Brownian path, however, is intractable in practice.

As an efficient alternative, Walk-on-Spheres (WoS) [Muller 1956] leverages the *Mean Value Property* of harmonic functions. For any ball $B(x)$ centered at x, a Brownian motion started at x has the same probability of exiting at any point on $\partial B(x)$. WoS exploits this fact by repeatedly constructing the largest ball that is fully contained in Ω and jumping to a uniformly random point on its surface. The walk terminates once it arrives within a small tolerance $\epsilon > 0$ of $\partial\Omega$, after which the nearest boundary point is returned as the exit location. WoS converges to $\partial\Omega$ in $O(\log 1/\epsilon)$ time for sufficiently smooth boundaries [Binder and Braverman 2012].

4 Monte Carlo Estimation of DtN Operators

Our goal is to recover the low-frequency Steklov eigenspectrum of a triangle mesh $\mathcal{M}$ with vertices $\mathbf{V}$ and faces $\mathbf{F}$, which approximates an underlying boundary $\partial\Omega$. Rather than obtaining the spectrum directly, we estimate—with Monte Carlo—the Dirichlet-to-Neumann operator, whose leading eigenpairs encode the Steklov modes of interest. We begin by deriving a straightforward Monte Carlo estimator w.r.t. a discrete boundary mesh to make the basic idea concrete. This naïve estimator is impractical, so we propose an alternative formulation in terms of a jump-kernel identity that yields a practical DtN estimator, and hence the Steklov eigenspectrum. Our estimator yields a discrete operator that is guaranteed to be PSD (regardless of Monte Carlo noise) and avoids materializing large, dense intermediate matrices. We additionally extend to the exterior DtN problem, which is accelerated with a Kelvin transform.

Algorithm 1. Naïve Estimator for Interior DtN

Input: Boundary mesh $\mathcal{M}$, inset distance ε, sample count N
Output: Monte Carlo estimate $\widehat{\mathbf{S}} \in \mathbb{R}^{V\times V}$ of the DtN stiffness matrix

$\widehat{\mathbf{S}} \leftarrow \mathbf{0} \in \mathbb{R}^{V\times V}$
for $m = 1, \ldots, N$ **do**
  $s,\ \mathrm{tri}_s \leftarrow \textsc{UniformSampleMesh}(\mathcal{M})$
  $x \leftarrow s - \varepsilon\, n_s$ — *Inset evaluation point*
  $t,\ \mathrm{tri}_t \leftarrow \textsc{WalkOnSpheres}(x)$ — *Sampled termination point*
  $b_s \leftarrow \textsc{BarycentricCoords}(s, \mathrm{tri}_s)$ — *Known,* $(\mathtt{V},)$
  $b_t \leftarrow \textsc{BarycentricCoords}(t, \mathrm{tri}_t)$ — *Sampled,* $(\mathtt{V},)$
  $\widehat{\mathbf{S}} \leftarrow \widehat{\mathbf{S}} + \frac{|\partial\Omega|}{N\varepsilon}\, b_s(b_s - b_t)^\mathsf{T}$ — *Sparse rank-1 update,* $(\mathtt{V},\mathtt{V})$
end for
$\widehat{\mathbf{S}} \leftarrow \frac{1}{2}(\widehat{\mathbf{S}} + \widehat{\mathbf{S}}^\mathsf{T})$ — *Make empirical operator symmetric*
return $\widehat{\mathbf{S}}$

4.1 A Naïve Estimator

The most straightforward approach to estimate Λ with Monte Carlo is to write the directional derivative, ∂_n, in Eq. 11 as a boundary limit

$$\mathcal{E}[f,g] = \int_{\partial\Omega} f(s) \cdot \left(\lim_{\varepsilon\to 0} \frac{g(s) - u_g(s - \varepsilon n_s)}{\varepsilon} \right) ds. \tag{13}$$

We write the Monte Carlo estimator in terms of a boundary point s sampled from the uniform distribution $U(\partial\Omega)$. From s, we estimate the normal derivative of the harmonic extension by launching a WoS sample from an inset point $x(s) = s - \varepsilon n_s$, where n_s is the outward surface normal and ε is a chosen distance used to approximate the derivative. Then, the bilinear form can be written as the expectation

$$\mathcal{E}[f,g] \approx \mathbb{E}_{s\sim U(\partial\Omega), t\sim\omega^\Omega_{x(s)}} \left[|\partial\Omega|\, f(s)\, \frac{g(s) - g(t)}{\varepsilon} \right]. \tag{14}$$

To discretize this expression, let $\mathbf{\Phi} = \{\phi_1, \phi_2, \ldots, \phi_V\}$ denote the standard piecewise-linear hat basis on $\mathcal{M}$. The discrete bilinear form is represented by the stiffness matrix $\mathbf{S} \in \mathbb{R}^{V\times V}$ with entries

$$S_{ij} \approx \frac{|\partial\Omega|}{\varepsilon}\, \mathbb{E}_{s\sim U(\partial\Omega),\, t\sim\omega^\Omega_{x(s)}} \left[\phi_i(s) \big(\phi_j(s) - \phi_j(t) \big) \right]. \tag{15}$$

If $b(p) = (\phi_1(p), \ldots, \phi_V(p))^\mathsf{T}$ denotes the vector of hat-function values at a boundary point p, then we can equivalently write the following expression for the entire stiffness matrix $\mathbf{S}$

$$\mathbf{S} \approx \frac{|\partial\Omega|}{\varepsilon}\, \mathbb{E}_{s\sim U(\partial\Omega),\, t\sim\omega^\Omega_{x(s)}} \left[b(s) \big(b(s) - b(t) \big)^\mathsf{T} \right]. \tag{16}$$

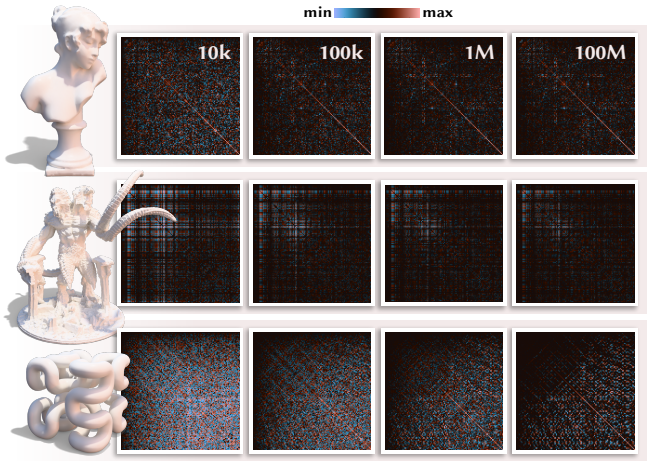

Fig. 5. Progression of our interior DtN estimator over several Monte Carlo sample counts. Operators are represented in a Galerkin basis of size 128.

Here $b(p)$ is the sparse vector of barycentric coordinates of p in its corresponding triangle. Thus, a WoS sample contributes a sparse rank-one update supported only on the vertices of the source and exit triangles. Averaging updates over N samples yields the empirical operator $\widehat{\mathbf{S}}$. We summarize this procedure in Algorithm 1.

While conceptually straightforward, this estimator suffers from two deficiencies that render it impractical for downstream tasks:

(1) **PSD only in expectation.** The Steklov matrix $\mathbf{S}$ is symmetric positive semidefinite (see Eq. 12); however, the straightforward estimator in Eq. 16 is only PSD *in expectation*, specifically because the outer product is taken over generally distinct vectors. A finite-sample estimate can easily violate this condition, and in practice the estimated matrix frequently has negative eigenvalues, which corrupts the behavior of downstream algorithms.

(2) **High variance.** This estimator inherits a well-known weakness of Monte Carlo estimators for differential quantities: approximating a normal derivative by a finite-difference quotient is noisy, and the variance blows up as ε is decreased to reduce bias. In addition, choosing a globally-valid inset distance ε is itself delicate on meshes with thin features, where even a modest inset can leave the domain and lead to incorrect MC samples.

The first issue is a fundamental shortcoming of the bilinear form (Eq. 13) from which the naïve estimator was derived. Below, we proceed with a key insight of this paper: the estimator can instead be derived from an alternative representation of the DtN operator, an expectation over rank-1 PSD outer products, yielding a PSD estimate. Further, we show that this representation can be decomposed into analytical and empirical parts, substantially reducing variance.

4.2 A Jump-Kernel Decomposition

The Beurling-Deny Theorem [Beurling and Deny 1958] asserts that regular[1] Dirichlet forms decompose into three canonical parts: a *strongly-local* (diffusion) term, a *jump* term, and a *killing* term. For

[1] *Regular* implies additional constraints on the Dirichlet form's domain and underlying topology, see Section 1.1 of Fukushima et al. [1994].

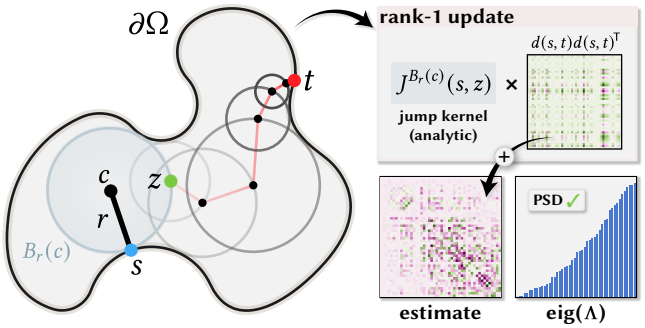

Fig. 6. **Our final interior Dirichlet-to-Neumann estimator.** A point $s \sim U(\partial\Omega)$ is sampled uniformly from the surface, from which we compute its largest tangent ball $B_r(c)$ with radius r centered at c. The integral decomposition in Section 4.2 allows us to write the DtN estimator w.r.t. points z drawn uniformly on $\partial B_r(c)$, effectively factoring out the analytic jump kernel $J^{B_r(c)}(s,z)$ from the boundary integral. A Walk-on-Spheres sample is cast from z, terminating at surface point t, which completes the rank-one matrix update: $J^{B_r(c)}(s,z) \cdot d(s,t)\, d(s,t)^\mathsf{T}$. Our utilization of the Beurling-Deny Theorem ensures the matrix update is positive semidefinite, and hence the Monte Carlo estimate robustly recovers the structure of the Steklov eigenspectrum. Pseudocode for this procedure is in Algorithm 2.

the DtN Dirichlet form $\mathcal{E}$, which arises as the trace form of volumetric Dirichlet energy (Eq. 10), only the jump component survives. Concretely, as shown by Chen and Fukushima [2011, Eq. 5.8.4], there exists a nonnegative symmetric kernel $J^\Omega : \partial\Omega \times \partial\Omega \to \mathbb{R}_{\geq 0}$ such that

$$\mathcal{E}[f,g] = \frac{1}{2} \iint_{\partial\Omega\times\partial\Omega} \big(f(s) - f(t)\big)\big(g(s) - g(t)\big)\, J^\Omega(s,t)\, ds\, dt. \quad (17)$$

This identity decomposes the DtN's Dirichlet energy form—originally a volume integral (Eq. 12)—into a symmetric boundary-to-boundary process that we can sample. We call J^Ω the *jump kernel* of Λ, since it governs how the associated boundary process jumps between distant surface points. Translating the work of Chen and Fukushima [2011, Eq. 5.8.2] into our conventions, the jump kernel,

$$J^\Omega(s,t) = -\partial_{n_s} P^\Omega(s \to t), \quad (18)$$

is the inward normal derivative of the Poisson kernel. This equation should be understood as a boundary limit, where

$$-\partial_{n_s} P^\Omega(s \to t) := \lim_{\epsilon \to 0} \frac{1}{\epsilon} P^\Omega(s - \epsilon n_s \to t) \quad (19)$$

because $G^\Omega(s,t)$ and $P^\Omega(s \to t)$ are both zero when $s, t \in \partial\Omega$, and $P^\Omega(s \to t)$ has a singularity when $s = t$.

Guaranteeing a PSD Estimate. To see why the Beurling-Deny decomposition enables a PSD-by-construction estimator, let us again consider a set of boundary basis functions, $\{\phi_k\}_{k=1}^K$, which can be evaluated at points $s \in \partial\Omega$. The Steklov stiffness matrix $\mathbf{S} \in \mathbb{R}^{K\times K}$ with entries $\mathbf{S}_{ij} = \mathcal{E}[\phi_i, \phi_j]$ can be expressed via Eq. 17 as

$$\mathbf{S} = \frac{1}{2} \iint_{\partial\Omega\times\partial\Omega} d(s,t)\, d(s,t)^\mathsf{T} J^\Omega(s,t)\, ds\, dt, \quad (20)$$

where $d(s,t) := b(s) - b(t) \in \mathbb{R}^K$ is the difference vector between the basis evaluations at two boundary points s and t. The integrand,

$d(s,t)\,d(s,t)^\mathsf{T} J^\Omega(s,t)$, is the product of a rank-one PSD matrix and a non-negative scalar, and hence is itself PSD. What remains is to sample pairs (s,t) according to a distribution related to the jump kernel $J^\Omega(s,t)$. A direct approach to sample the jump kernel is to estimate the differential Poisson kernel $\partial_{n_s} P^\Omega(s \to t)$ using finite-differences, analogous to the derivation of the naïve estimator. As discussed in Section 4.1, this approach incurs extreme variance and is brittle when shapes have thin structures.

Variance Reduction. Instead of estimating the jump kernel with finite differences, we exploit the fact that the jump kernel is analytically known on a ball (see Section C.2). By applying the analysis of Yu et al. [2024], we are able to write a variance-reduced estimator of Eq. 20 by decomposing $J^\Omega(s,t)$ into two quantities: a closed-form jump kernel for the largest interior ball tangent to s, and the result of WoS samples launched from points on the surface of this ball. This decomposition effectively factors out analytically-known jump kernel behavior, casting WoS samples solely for the nontrivial component of the jump kernel that cannot be known ahead of time.

Fix a boundary point $s \in \partial\Omega$ and let $B_r(c) \subset \Omega$ be the largest ball contained in Ω that is tangent to $\partial\Omega$ at s. The center of this tangent-ball lies at a point $c = s - r n_s$ offset along the inward normal by the ball radius r. A Brownian motion starting at a point $x \in B_r(c)$ must first exit the ball at some point $z \in \partial B_r(c)$ before eventually exiting the larger domain Ω at a point $t \in \partial\Omega$. Therefore, we can factor the walk into two independent sub-walks based on these two exit points using the *Strong Markov Property* of Brownian motion,

$$P^\Omega(x \to t) = \int_{\partial B_r(c)} P^{B_r(c)}(x \to z)\, P^\Omega(z \to t)\, dz. \tag{21}$$

The first factor,

$$P^{B_r(c)}(x \to z) = \frac{1}{4\pi r} \frac{r^2 - |x-c|^2}{|x-z|^3}, \tag{22}$$

is the classical Poisson kernel for the ball, known in closed form. The jump kernel for the ball is the normal derivative of its Poisson kernel (Eq. 18), so for $s \neq z$ we get the analytical representation

$$J^{B_r(c)}(s,z) = \frac{1}{2\pi |s-z|^3}. \tag{23}$$

By writing the jump kernel in terms of the Poisson kernel (Eq. 18) and using the Strong Markov Property (Eq. 21), we obtain

$$J^\Omega(s,t) = \int_{\partial B_r(c)} J^{B_r(c)}(s,z)\, P^\Omega(z \to t)\, dz, \tag{24}$$

proving that we only need to simulate random walks starting from the tangent-ball surface (see Appendix Section C.2 for derivations).

Improved DtN Estimator. Substituting this jump kernel decomposition into the Beurling-Deny representation (Eq. 20) yields

$$\mathrm{S} = \frac{1}{2} \iint_{\partial\Omega\times\partial\Omega} d(s,t)\, d(s,t)^\mathsf{T} \left(\int_{\partial B_r(c)} J^{B_r(c)}(s,z)\, P^\Omega(z \to t)\, dz \right) dt\, ds,$$

which equivalently rearranges as

$$\mathrm{S} = \frac{1}{2} \iint_{\partial\Omega\times\partial B_r(c)} \left(\int_{\partial\Omega} d(s,t)\, d(s,t)^\mathsf{T} P^\Omega(z \to t)\, dt \right) J^{B_r(c)}(s,z)\, dz\, ds.$$

Algorithm 2. Final Interior DtN Estimator

Input: Boundary mesh $\mathcal{M}$, Galerkin basis Φ, sample count N
Output: PSD estimate $\widehat{\mathrm{S}} \in \mathbb{R}^{K\times K}$ of the DtN stiffness matrix

```
Ŝ ← 0 ∈ R^{K×K}
for m = 1, ..., N do
    ▷ Find largest inscribed tangent ball B_r(c) at sampled point s
    s, tri_s ← UniformSampleMesh(M)
    r ← LargestTangentBall(s, n_s, r_max)
    c ← s − r · n_s
    ▷ Sample random walk starting from tangent ball surface
    z ← SampleBallSurface(c, r, n_s)
    J ← BallJumpKernel(s, z)
    t, tri_t ← WalkOnSpheres(z)
    ▷ Estimate flux of extended basis functions from t through s
    d ← 0 ∈ R^K
    for k = 1, ..., K do
        ▷ Evaluate k-th basis function at s and t
        b_s ← BarycentricInterp(s, tri_s, φ_k)
        b_t ← BarycentricInterp(t, tri_t, φ_k)
        ▷ Store estimated flux of k-th basis function
        d_k ← b_s − b_t
    end for
    ▷ Update Ŝ with rank-one PSD sample
    Ŝ ← Ŝ + (|∂Ω|·|∂B_r(c)| / 2N) · d d^T · J
end for
return Ŝ
```

Recognizing the parenthetical as a Poisson integral (Eq. 4) allows us to rewrite it as an expectation (Eq. 5)

$$\mathrm{S} = \frac{1}{2} \iint_{\partial\Omega\times\partial B_r(c)} \mathbb{E}_t \left[d(s,t)\, d(s,t)^\mathsf{T} \right] J^{B_r(c)}(s,z)\, dz\, ds, \tag{25}$$

where $t \sim \omega_z^\Omega$ is the exit point of a Brownian motion started from $z \in \partial B_r(c)$ on the boundary of the tangent-ball. Finally, we can again write the corresponding Monte Carlo estimator in terms of $s \sim U(\partial\Omega)$ and $z \sim U(\partial B_r(c))$ sampled uniformly.

$$\mathrm{S} = \mathop{\mathbb{E}}_{s,z,t} \left[\frac{|\partial\Omega|\,|\partial B_r(c)|}{2}\, d(s,t)\, d(s,t)^\mathsf{T} J^{B_r(c)}(s,z) \right] \tag{26}$$

Algorithm 2 details the pseudocode for our final DtN estimation procedure, which is illustrated in Figure 6.

4.3 Exterior DtN Operators

The DtN operator we have estimated thus far is defined with respect to the *interior* of a given closed boundary. For shapes that are composed of multiple disjoint bounding surfaces—e.g. a mesh of a person holding an apple—these boundaries do not interact, as the Brownian motions simulated in either domain cannot terminate on the opposing surface. To complement our interior DtN operator, we consider its *exterior* variant, defined by an equivalently posed Dirichlet problem on the unbounded exterior domain $\Omega_{\text{ext}} := \mathbb{R}^3 \setminus \overline{\Omega}$ (i.e., the complement of Ω sharing the same boundary) with the additional Dirichlet condition: $u(x) \to 0$ as $|x| \to \infty$. The resulting operator, denoted Λ_{ext}, is the *exterior DtN operator*, which treats

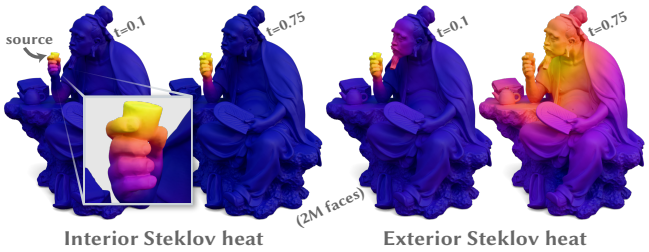

Fig. 7. Lu Yu enjoying a hot cup of tea. The exterior Steklov heat warms his face through the ambient volume around the surface. The interior heat evolves slowly by comparison, due to the thin cavity of his hand.

the unbounded exterior as a single unified volume (see Figure 7), allowing it to capture boundary-to-boundary interactions between surfaces that otherwise enclose completely separate shapes.

Escape to Infinity. A Brownian motion started in a bounded interior domain $\Omega \in \mathbb{R}^3$ is guaranteed to hit the boundary at some point in time. For a Brownian motion in the unbounded exterior domain Ω_{ext}, however, one of two things can happen:

(1) it *hits* the boundary and terminates.

(2) it *escapes* and never returns.

In general, it is difficult to Monte Carlo sample first-passage points in the exterior domain because random walk length is unbounded; no matter how far a particle is from the surface, it always has a nonzero probability of returning to the boundary.

Exterior DtN. The Beurling-Deny formula decomposes these two possibilities—that a random walk hits the boundary or escapes—as two components of a single boundary process. Unlike the interior operator, which only has a jump term (Eq. 17), the exterior operator is composed of both a jump term and a *killing* term [Chen and Fukushima 2011, Eq. 5.8.9]:

$$\mathcal{E}_{\text{ext}}[f,g] = \frac{1}{2}\iint_{\partial\Omega\times\partial\Omega} \big(f(s)-f(t)\big)\big(g(s)-g(t)\big)\, J^{\Omega_{\text{ext}}}(s,t)\, ds\, dt + \int_{\partial\Omega} f(s)\, g(s)\, K^{\Omega_{\text{ext}}}(s)\, ds. \tag{27}$$

We call $K^{\Omega_{\text{ext}}} : \partial\Omega \to \mathbb{R}$ the *killing measure* of the exterior domain Ω_{ext} because it models the local rate at which the associated boundary process is killed by escaping to infinity. Chen and Fukushima [2011, Eq. 5.8.6] define $K^{\Omega_{\text{ext}}}$ in terms of *escape probability*,

$$q(x) := 1 - \mathcal{H}^{\Omega_{\text{ext}}}[\mathbf{1}](x). \tag{28}$$

Then, the killing measure is the normal derivative of this probability,

$$K^{\Omega_{\text{ext}}}(s) = \partial_{n_s} q(s). \tag{29}$$

A faithful estimator of Λ_{ext} must accurately model both the exterior jump kernel $J^{\Omega_{\text{ext}}}$ and killing measure $K^{\Omega_{\text{ext}}}$, which entails launching random walks into the infinite void. Handling this unboundedness in an unbiased and sample efficient way remains challenging without a shift in perspective.

The Kelvin Transform. The obstacle to Monte Carlo sampling $J^{\Omega_{\text{ext}}}$ and $K^{\Omega_{\text{ext}}}$ is the lack of termination guarantees for random walks in the exterior domain. This issue is rectified using a change of coordinates introduced to the graphics community by Nabizadeh et al. [2021]: the *Kelvin transform* is a conformal mapping $\kappa : \mathbb{R}^3 \cup \{\infty\} \to \mathbb{R}^3 \cup \{\infty\}$ sending $x \mapsto x/|x|^2$ (with $0 \mapsto \infty$ and $\infty \mapsto 0$). Assuming $0 \in \Omega$ (which can always be arranged by translation), define $\Omega^\star_{\text{ext}} := \kappa(\Omega_{\text{ext}} \cup \{\infty\})$. Under the Kelvin transform, $\Omega^\star_{\text{ext}}$ is a *bounded* domain in $\mathbb{R}^3$ that includes the origin (as the image of ∞).

Sampling Exterior Quantities. The Kelvin transform is exceptionally useful in the harmonic exterior setting. In 3D, a function $u(x)$ (with decay at infinity) is harmonic at $x \in \Omega_{\text{ext}}$ if and only if the function $\kappa[u](x^\star) := |x|\, u(x)$ is harmonic at $x^\star \in \Omega^\star_{\text{ext}}$, where $x^\star := \kappa(x)$ denotes the image of x under the Kelvin transform (see Appendix Section C.3). Ultimately, this correspondence gives rise to proxies for the exterior domain jump kernel

$$J^{\Omega_{\text{ext}}}(s,t) = |s^\star|^3\, |t^\star|^3\, J^{\Omega^\star_{\text{ext}}}(s^\star, t^\star) \tag{30}$$

and the killing measure

$$K^{\Omega_{\text{ext}}}(s) = 4\pi\, |s^\star|^3\, P^{\Omega^\star_{\text{ext}}}(0 \to s^\star), \tag{31}$$

both of which are written in terms of the bounded domain $\Omega^\star_{\text{ext}}$ where Walk-on-Spheres regains its termination guarantees. The identities in Eq. 30 and Eq. 31 are derived in Appendix Sections C.5 and C.6 respectively.

An Exterior DtN Estimator. Because $J^{\Omega^\star_{\text{ext}}}$ is an *interior* jump kernel on a bounded domain, it admits the same tangent-ball decomposition used in Eq. 24. Then, the very same integral rearrangement used in Section 4.2—this time applied to the exterior Beurling-Deny formula for DtN (Eq. 27)—produces an exterior estimator that operates in the Kelvin-transformed domain.

$$\mathrm{S}_{\text{ext}} = \mathop{\mathbb{E}}_{s,z^\star,t^\star}\left[\frac{|\partial\Omega|\,|\partial B_r(c^\star)|\,|s^\star|^3}{2\,|t^\star|}\, d(s,t)\, d(s,t)^{\mathsf{T}}\, J^{B_r(c^\star)}(s^\star, z^\star)\right] + \mathop{\mathbb{E}}_{t_0^\star}\left[\frac{4\pi}{|t_0^\star|}\, b(t_0)\, b(t_0)^{\mathsf{T}}\right] \tag{32}$$

The WoS hit points $t^\star \sim \omega^{\Omega^\star_{\text{ext}}}_{z^\star}$ and $t_0^\star \sim \omega_0^{\Omega^\star_{\text{ext}}}$, for the jump term and killing term respectively, are obtained by casting a Brownian motion from a tangent ball sample $z^\star \sim U(\partial B_r(c^\star))$ and $0 \in \Omega^\star_{\text{ext}}$ in the inverted domain. Surface points $s \sim U(\partial\Omega)$ are still sampled uniformly on the *primal* boundary. A derivation of the exterior estimator is given in Appendix Section C.7.

4.4 Boundary function spaces

To recover the first K Steklov eigenmodes, we observe that it suffices to work with a discrete representation of the DtN operator that is significantly smaller than the full $V \times V$ dense matrix. We directly estimate a $K \times K$ reduced DtN operator that acts within a projected space $\mathbb{R}^K \subseteq \mathbb{R}^V$. Hence, we follow the *Ritz-Galerkin method* to approximate the true Steklov eigenvalues and eigenvectors using an orthonormal basis $\{\phi_1, \phi_2, \ldots, \phi_K\}$ defined on $\partial\Omega$. Because the estimators derived in the preceding sections are written only in terms

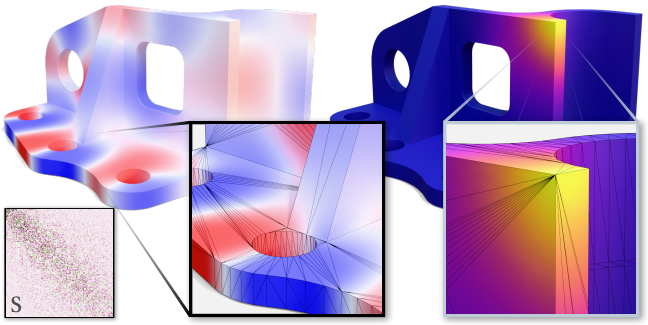

Fig. 8. Our point-based estimator fully decouples operator resolution from that of the surface discretization, making it robust to exceedingly poor triangulations. By contrast, traditional methods require retriangulation to operate in this regime, which is i) expensive, ii) not robust when the shape is multi-component, and iii) may degrade surface geometry.

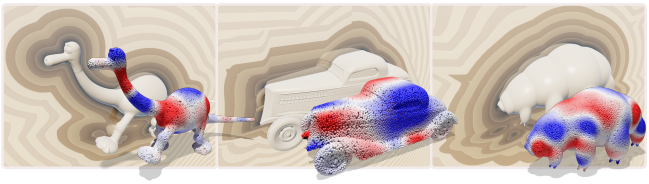

Fig. 9. Our point-based DtN estimator can be applied directly to signed distance fields without intermediate meshing. We visualize Steklov eigenfunctions on their respective point cloud function spaces.

of evaluations of a finite boundary basis—i.e. $b(s)$ and $d(s,t)$—we consequently need not restrict ourselves to Galerkin bases induced by mesh connectivity (e.g. the ordinary piecewise-linear function space). Any compact basis on $\partial\Omega$ can be used, provided it can be evaluated at arbitrary points.

This separation motivates the use of an alternative function space—one that lives on *points* sampled on the surface—that is completely decoupled from the surface discretization itself, which can be poor for in-the-wild geometry. We first discuss our procedure when operating under a mesh-based function space, then we expand on the point-based alternative.

Mesh-based Galerkin basis. Our default choice for the Galerkin basis Φ is the low-frequency eigenmodes of the cotangent Laplacian defined on the boundary mesh. Basis evaluation, $b(s)$, is performed by barycentric interpolation in each triangle. This choice is efficient and works well even when the boundary mesh has multiple connected components, despite the Laplacian (and its eigenmodes) being decoupled. Our estimated DtN operators nevertheless are able to couple components through the volumetric domain.

Point-based Galerkin basis. For meshes with extremely poor surface triangulation, it is in some sense hopeless to represent interesting functions directly in the linear elements themselves (i.e. without retriangulation or similar interventions). The flexibility of Walk-on-Spheres allows us to separate the *geometry* that governs Brownian motions from the *function space* in which we represent our solutions [Sawhney and Crane 2020]. With this fact in mind, we opt to define a function space on a set of densely sampled points on the

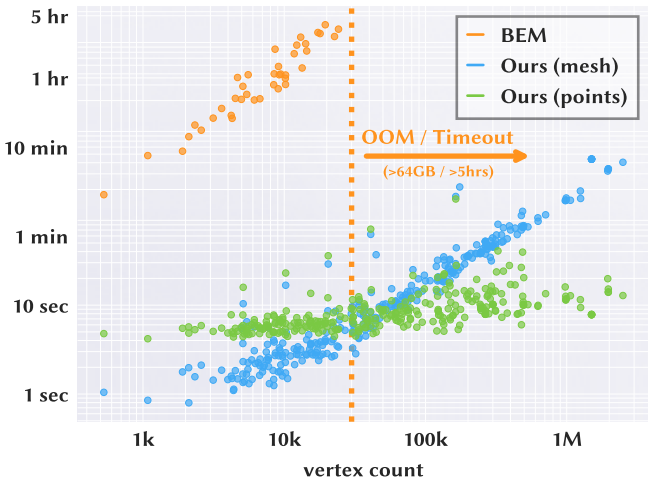

Fig. 10. **Runtime of methods for computing the first 128 interior Steklov eigenmodes.** Our method (10M Monte Carlo samples) is orders of magnitude faster than the BEM-based alternative. Galerkin basis precomputation is included in runtime, making the fixed-size point representation fastest on average— 2^{14} points were used. Out-of-memory (OOM) and runtime constraints precluded running BEM beyond moderate vertex counts.

mesh surface. We sample points $P = \{p_i\}_{i=1}^{N_p}$ on $\partial\Omega$ using blue noise sampling [Bridson 2007], and compute the low-frequency eigenmodes of the associated point cloud Laplacian [Sharp and Crane 2020], which serve as the boundary basis for this representation. During Monte Carlo sampling, WoS paths still terminate on the input mesh, and we interpolate basis values from the sampled points P to the termination point using a k-NN interpolator. In particular, we consider a bilateral weighting on both Euclidean distance and normal angular deviation over the k nearest point samples (see Appendix B for details). Replacing barycentric interpolation with this evaluator leaves the estimator unchanged. This scheme can be applied to other surface representations as well, e.g. signed distance fields (see Figure 9).

5 Implementation

We implement our estimators in CUDA, using `cuBQL` BVH [Wald 2026] as our only major dependency for WoS distance queries. In the exterior domain, we model inverted triangle meshes with spherical-triangular elements and employ appropriate (non-planar) distance queries to these elements. Unless otherwise specified, we forgo sampling of the killing measure—effectively fixing it to zero—which allows the exterior operator to behave conservatively[2]. We use a WoS termination tolerance of `1e-6` for all experiments. Tangent balls are computed using the bisection algorithm of Yu et al. [2024] with 10 iterations; similarly, we perform antithetical sampling on each tangent ball. We importance sample tangent balls according to their jump kernels and forgo sampling points on the ball especially close to the surface point s to avoid the singularity. Namely, we

[2]The killing measure is the only Beurling-Deny term that acts on constants. Dropping it restores a constant zero eigenmode, making the associated heat flow preserve constants instead of dissipating functions to zero, as is typical for diffusion operators used in geometry processing.

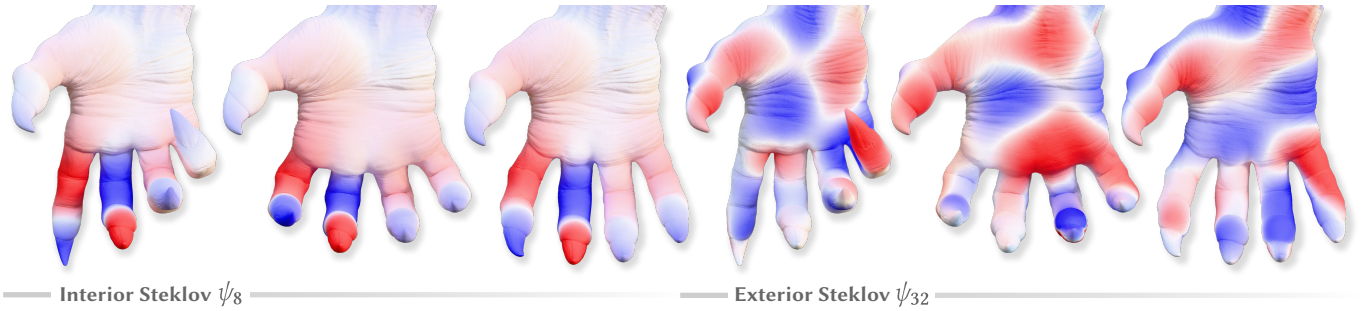

Fig. 11. Our interior and exterior Steklov estimator applied to an extremely high-resolution articulating hand with ∼4 million faces. The interior eigenfunctions are relatively stable under minor deformations; whereas the exterior counterparts are more sensitive to relative positions of the fingers and palm.

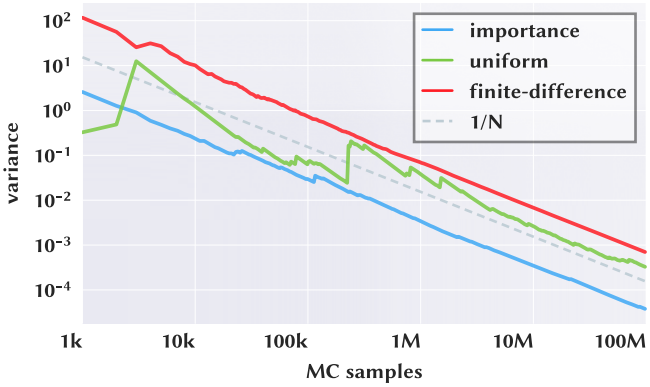

Fig. 12. **Mean matrix-entry variance of our estimated DtN operator.** Our estimators converge under the usual $O(1/N)$ variance w.r.t. the number of Monte Carlo samples. The finite-difference estimator (red) exhibits highest variance across all sample counts. Employing the integral decomposition in Section 4.2 and uniformly sampling the surface of tangent balls reduces variance (green). Importance sampling the jump kernel on these balls further reduces variance and stabilizes the estimator (blue).

exclude a spherical cap subtending $\sim 2.5^\circ$ about s from the domain of integration. Our CUDA kernels sort sample points $s \in \partial\Omega$ via Z-order curves, which greatly reduces warp divergence; additionally, the outer product $d(s,t)\, d(s,t)^T$ is deferred to be computed in batches to avoid shared memory contention of the empirical operator $\mathbf{S}$. Finally, we note that our theoretical treatment in Section 4 is with respect to much stricter conditions (i.e. smooth, closed domains) than the applied setting we proceed with—we refer to Appendix A for clarifying details surrounding this practical gap.

6 Evaluation

We compare our Monte Carlo estimators to the method of Wang et al. [2018] (denoted `BEM`), which employs a matrix-free boundary element method to resolve interior Steklov eigenmodes. We first demonstrate that our interior DtN estimator yields a matrix whose eigenspectrum faithfully converges to that of BEM. We then explore the unique scalability and robustness of our method, showing that it can be rapidly applied to large meshes and ones with complex or degenerate surface discretizations.

Setup. We apply the open source implementation of Wang et al. [2018] to compute the first $k = 128$ interior Steklov eigenmodes on shapes sourced from Thingi10k [Zhou and Jacobson 2016], using 25 LOBPCG iterations. Data that is labeled as `preprocessed` implies that the corresponding shapes have undergone retriangulation using fTetWild [Hu et al. 2020] surface extraction—we found this necessary to ensure BEM converges on shapes with poor discretizations. We employ our Monte Carlo estimator on the same shapes, using a Galerkin basis proportionally sized to k—in practice, we use $k + 32$ basis functions. We evaluate our mesh-based and point-based variants as defined in Section 4.4. Reported runtimes for our method are recorded w.r.t. a consumer-grade NVIDIA RTX 4090.

Convergence. We establish the correctness of our estimator by directly comparing its eigenmodes to BEM on preprocessed Thingi10k shapes. In particular, Figure 13 compares the eigenvalues of both methods across 20 shapes. Comparing the *eigenvectors* themselves is less trivial due to sign and permutation ambiguity; further, repeated eigenvalues are valid under arbitrary orthogonal rotations. For this reason, we opt to compare eigenvectors through basis-invariant quantities induced by the corresponding spectral decompositions. In particular, Figure 14 compares the Heat Kernel Signatures (HKS) computed using both spectra, showing that—up to subtle numerical differences—their structure is nearly identical. Finally, Figure 12 verifies that our estimators converge under $O(1/N)$ variance in the number of samples, as expected of Monte Carlo methods; the integral decomposition of Section 4.2 specifically vastly reduces variance as compared to the finite-difference baseline.

Robustness. Our DtN estimators inherit the robustness of Monte Carlo PDE solvers, making them directly applicable to shapes with hundreds of connected components, degenerate triangulations, and self-intersections. In Figure 8 we show that our construction is directly applicable to a CAD-like model that is primarily composed of sliver triangles; in spite of this, the estimated Steklov modes are meaningful and produce correct heat-like behavior. Similarly, Figure 2 shows the behavior of our interior and exterior Steklov modes on a non-trivial `Commodore 64` mesh that contains 2662 connected components with poor element quality. The interior Steklov heat dissipates beneath the keycaps, diffusing slowly due to the disproportionate number of obstructions, compared to the exterior heat, which diffuses over a broader profile. We define *Steklov heat* as the

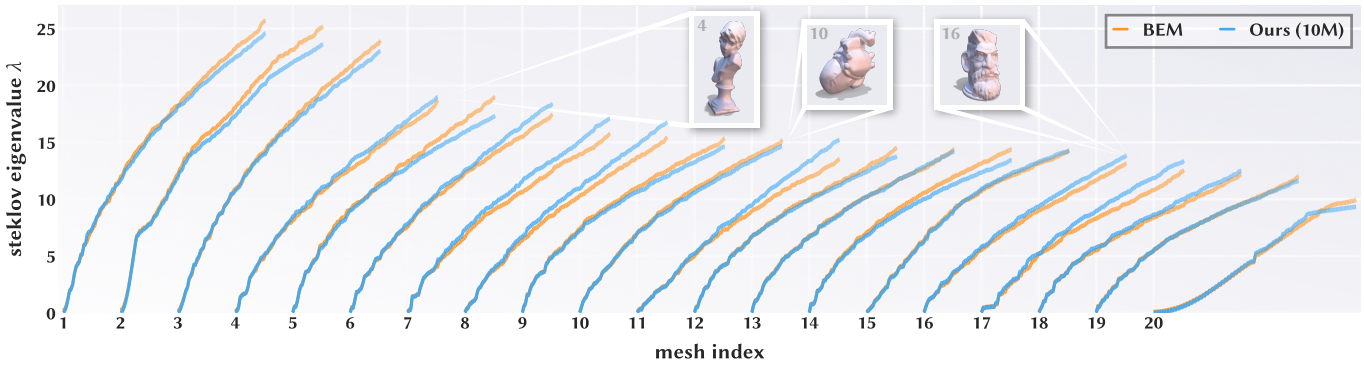

Fig. 13. Our estimated interior Steklov eigenspectra computed on 20 preprocessed shapes from Thingi10k, compared to the BEM-based method of Wang et al. [2018]. Each pair of eigenspectra represent a unique shape. Our method faithfully approximates the Steklov eigenmodes while reducing runtime massively. We note that BEM employs an iterative method that admits error, hence exact replication—especially near the tail-end of the spectrum—is not expected.

Fig. 14. Comparison of heat kernel signatures derived from the interior Steklov eigenspectra, as computed by our Monte Carlo estimator and BEM.

heat equation associated with the DtN operator [Wang et al. 2018],

$$\frac{du}{dt} = -\Lambda u, \tag{33}$$

whose solution is approximated in the Steklov eigenbasis as

$$u_t = \sum_{k=1}^{K} e^{-\lambda_k t} a_k \psi_k, \tag{34}$$

where (λ_k, ψ_k) are the k-th Steklov eigenpair, and $a_k = \langle u_0, \psi_k \rangle_{\mathbf{M}}$ represents spectral coefficients of the initial heat distribution. The DtN operator, Λ, can be taken as either the interior or exterior variant. To visualize the fundamental behavior of Steklov heat, Figures 2, 8, and 7 take their initial heat distributions, u_0, as a unit impulse localized at one vertex. Although Λ and its eigenmodes live on the boundary, the Steklov heat equation behaves volumetrically, and dissipates *through* the volume implied by a surface mesh.

Scalability. The CUDA implementation of our estimators is able to use modern GPUs effectively, making it substantially more efficient than traditional volumetric spectral methods, which are ordinarily CPU-bound. Figure 10 compares the runtime of our estimator at 10 million samples against that of BEM. We find that our method is orders of magnitude faster across all mesh sizes, and is able to handle much larger element counts, whereas BEM eventually fails due to out-of-memory errors or compute timeout (5 hours). Figure 17 shows timings for our interior and exterior estimators extended to the large-scale, uncurated Objaverse dataset [Deitke et al. 2023]. We discuss further details and findings for this experiment in Section 7—critically, scaling existing tetrahedral or BEM-based methods to the size and topological irregularity of this dataset is infeasible.

7 Representation Learning on Meshes

The preceding sections make interior and exterior Steklov eigenmodes available at the scale and irregularity of modern shape datasets. To demonstrate this fact further, and to highlight the utility of the Steklov modes themselves, we employ these spectra as primitive building blocks in a representation learning pipeline for meshes. In particular, we use Steklov operators to define volumetrically-informed feature transformations in a neural architecture, in similar style to previous (intrinsic) networks for meshes [Smirnov and Solomon 2021; Sharp et al. 2022; Maesumi et al. 2025].

Learning 3D shape representations with contrastive pre-training has been studied in several works, which can largely be categorized as point cloud-based networks, and ones that use multi-view images—we refer to Lee et al. [2025] for an overview. To the best of our knowledge, there has yet to be a faithful attempt at large-scale contrastive pretraining on meshes directly, simply because the necessary geometry processing machinery does not exist—the limitations of intrinsic methods preclude their application to the

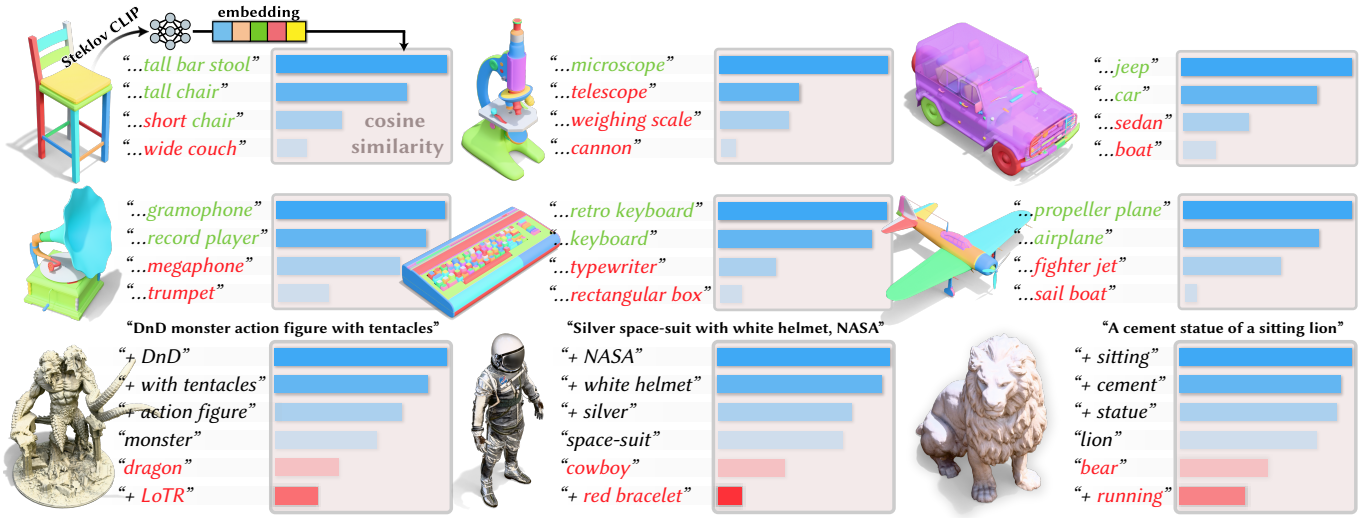

Fig. 15. **Qualitative results from our Steklov-CLIP model.** We probe representations learned by Steklov-CLIP on a range of meshes sourced from public repositories—all meshes shown have one or more of the following qualities: many connected components, poor element quality, or extremely dense triangulation (i.e. representative qualities of in-the-wild shapes). We probe our model by comparing cosine similarities of its shape embeddings to manually authored text queries. Green and red words indicate terms that are semantically relevant or dissimilar to the objects, respectively. The bottom row shows cosine similarity changing as more query terms are included. Similarity scores depicted as colored bars are drawn relative to the highest attained score.

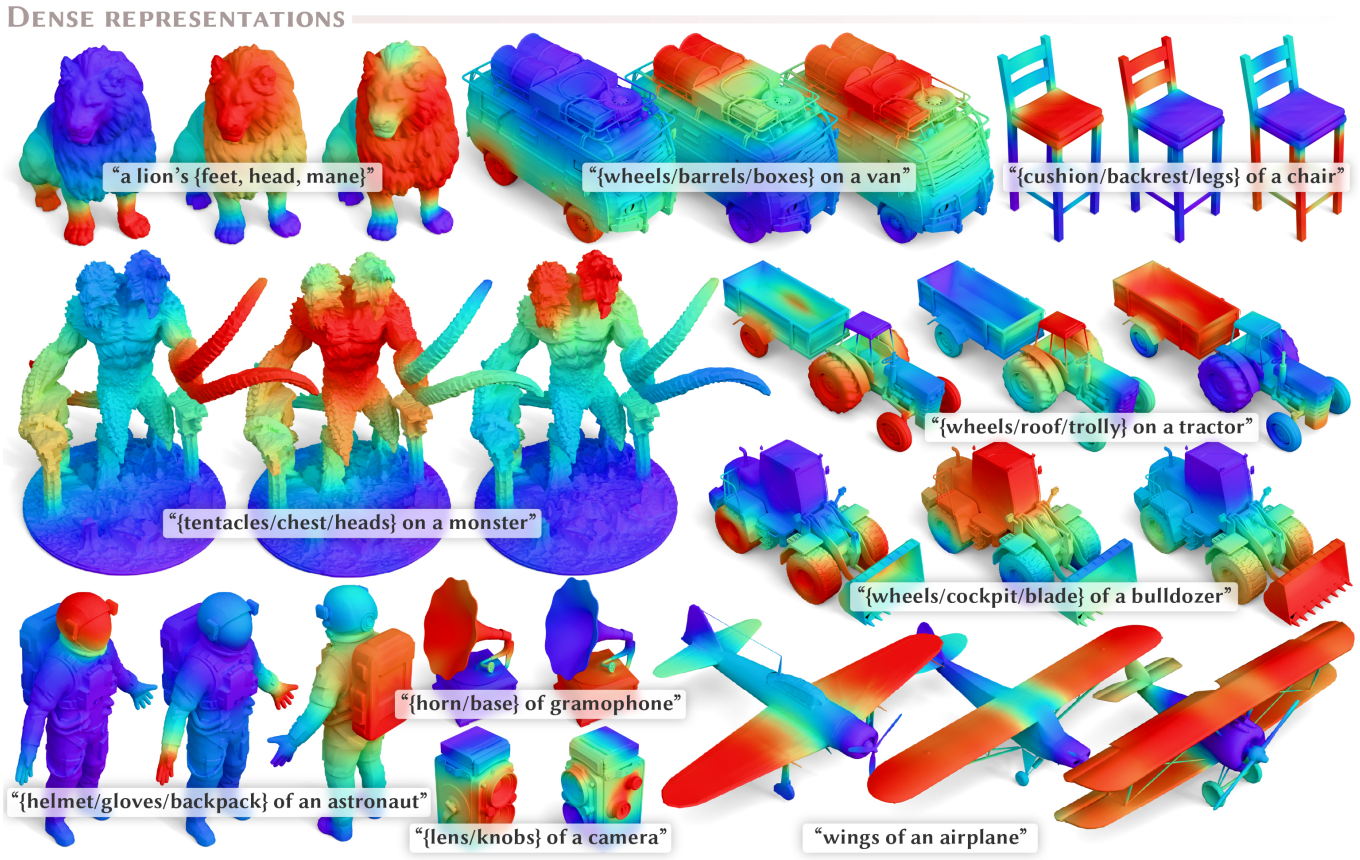

Fig. 16. **Saliency maps from our finetuned Steklov-CLIP.** We visualize cosine similarity of per-point embeddings produced by our model against text embeddings (overlaid). Our finetuned model is able to localize semantic parts, and even generalizes to heavy-tail cases (e.g. the two-headed Demogorgon).

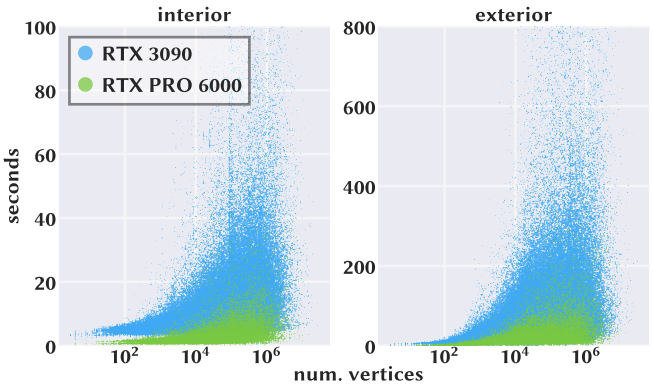

Fig. 17. Compute timings of our interior and exterior estimators (10M Monte Carlo samples) applied at scale to ∼450k shapes from Objaverse. Compute was distributed across a mix of RTX 3090 and RTX PRO 6000 GPUs. Our method effectively utilizes modern GPUs and hence we observe vastly better throughput on the latter (more modern) Blackwell chips. Further, the exterior estimator is generally slower due to Kelvin-inverted BVH distance queries, which are generally more expensive than ordinary triangular mesh queries.

kinds of datasets at hand, and traditional volumetric methods are too expensive to be applied at this scale. Our goal, then, is to train a mesh encoder whose embeddings lie in the same semantic space as a frozen text-image contrastive model (e.g. CLIP).

Preliminaries. Following the standard recipe in this domain, we consider a large collection of shapes obtained from the Objaverse dataset [Deitke et al. 2023], from which we have corresponding caption and multi-view image embeddings. In particular we use the embeddings provided by Lee et al. [2025], which employs the `laion2b_s34b_b79k` OpenCLIP checkpoint [Ilharco et al. 2021]. We precompute interior and exterior Steklov eigenspectra on ∼450,000 shapes as outlined in Section 6. We discard ∼ 30,000 shapes that were of extremely low quality. Our network is trained using an InfoNCE contrastive objective [Oord et al. 2018] defined over paired batches of embeddings from two modalities. Let $E_a = \{e_i^a\}_{i=1}^N$ and $E_b = \{e_i^b\}_{i=1}^N$, where $e_i^a, e_i^b \in \mathbb{R}^d$ denote embeddings of the i-th paired shape instance in modalities a and b. Then the one-way contrastive loss is

$$\ell_{a\to b}(E_a, E_b) = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{\exp\left(\langle e_i^a, e_i^b\rangle/\tau\right)}{\sum_{k=1}^{N}\exp\left(\langle e_i^a, e_k^b\rangle/\tau\right)}, \tag{35}$$

where τ is a learned temperature. We consider three modalities, yielding embeddings for shape S, text T, and images I, making the final symmetric objective

$$\mathcal{L}_{\text{CLIP}} = \tfrac{1}{4}\left(\ell_{S\to T} + \ell_{T\to S} + \ell_{S\to I} + \ell_{I\to S}\right). \tag{36}$$

Network Architecture. Our network architecture takes inspiration from DiffusionNet [Sharp et al. 2022] and Galerkin Transformer [Cao 2021]. More specifically, the former defines feature transformations through an intrinsic heat equation, solved via a spectral approximation—which takes the same form as Eq. 34—defined by the eigenspectrum of the Laplace-Beltrami operator. Our network employs a similar philosophy, though it uses interior and exterior Steklov spectra in place of this intrinsic alternative, which endows our network with the ability to propagate features through volumetric regions that are possibly disconnected. In particular, our core network block begins by applying Steklov heat filters to the incoming feature field, denoted $\mathbf{f} : \partial\Omega \to \mathbb{R}^C$. Letting $\mathbf{\Psi}_{\text{int}}, \mathbf{\Psi}_{\text{ext}}$ denote the interior and exterior Steklov bases, we apply per-channel

$$\mathbf{h}_s^c \leftarrow \mathbf{\Psi}_s \begin{bmatrix} e^{-\lambda_s^0 t_s^c} \\ \dots \\ e^{-\lambda_s^k t_s^c} \end{bmatrix} \odot (\mathbf{\Psi}_s^\mathsf{T}\mathbf{M}\,\mathbf{f}_s^c), \quad s \in \{\text{int}, \text{ext}\}. \tag{37}$$

For brevity we use $\cdot_s$ when applying operations with both Steklov spectra, and $\mathbf{t}_s \in \mathbb{R}^{C/2}$ is a learned per-channel heat time. Interior and exterior heat filters are applied to both halves of the feature vector, and the resulting features are concatenated back together.

Galerkin Transformer motivates a second ingredient in our architecture, though in a more indirect way. The relevance to our setting is in its observation that a softmax-free attention layer can be viewed as an operator assembled from inner products of functions, rather than a dense pair-wise interaction between tokens—or in our case, quadrature points. We employ this idea using Steklov eigenspaces as the underlying function space. In particular, we first apply a learned linear projection and then represent the resulting feature fields in the Steklov spectral domains,

$$\mathbf{u} = \mathbf{f}\,\mathbf{W}_{\text{in}}, \qquad \mathbf{Z}_{\text{int}} = \mathbf{\Psi}_{\text{int}}^\mathsf{T}\mathbf{M}\mathbf{u}, \quad \mathbf{Z}_{\text{ext}} = \mathbf{\Psi}_{\text{ext}}^\mathsf{T}\mathbf{M}\mathbf{u}. \tag{38}$$

A learned linear transformation then produces three coefficient fields, which represent query, key, and value matrices whose columns are scalar boundary functions represented in the Steklov spectrum

$$\mathbf{Q}_s = \mathbf{Z}_s\mathbf{W}_s^Q, \quad \mathbf{K}_s = \mathbf{Z}_s\mathbf{W}_s^K, \quad \mathbf{V}_s = \mathbf{Z}_s\mathbf{W}_s^V, \quad s \in \{\text{int}, \text{ext}\}. \tag{39}$$

We additionally modulate these functions by learned eigenvalue-dependent filters, in particular learned mixtures of heat kernels given by Eq. 34. Our block then forms the bilinear operators

$$\mathbf{G}_{\text{int}} = \tfrac{1}{k}\mathbf{K}_{\text{int}}^\mathsf{T}\mathbf{V}_{\text{int}}, \qquad \mathbf{G}_{\text{ext}} = \tfrac{1}{k}\mathbf{K}_{\text{ext}}^\mathsf{T}\mathbf{V}_{\text{ext}}, \tag{40}$$

where k is the size of our Steklov basis. In classical attention, this intermediate matrix (taken instead between queries and keys) would represent dense similarity between all quadrature points; here, however, $\mathbf{G}$ is a small operator on feature channels assembled by integrating over the Steklov function space, and notably its size is independent of the number of quadrature points on the surface.

We allow $\mathbf{G}_{\text{int}}$ and $\mathbf{G}_{\text{ext}}$ to exchange information through the learned update

$$\begin{bmatrix}\mathbf{G}_{\text{int}} \\ \mathbf{G}_{\text{ext}}\end{bmatrix} \leftarrow \begin{bmatrix}\alpha_{\text{ii}} & \alpha_{\text{ie}} \\ \alpha_{\text{ei}} & \alpha_{\text{ee}}\end{bmatrix}\begin{bmatrix}\mathbf{G}_{\text{int}} \\ \mathbf{G}_{\text{ext}}\end{bmatrix}, \tag{41}$$

where the coupling is learned independently for each attention head and initialized near the identity. Output coefficients are obtained by applying these operators to query vectors

$$\mathbf{Y}_{\text{int}} = \tfrac{1}{\sqrt{d}}\mathbf{Q}_{\text{int}}\mathbf{G}_{\text{int}}, \qquad \mathbf{Y}_{\text{ext}} = \tfrac{1}{\sqrt{d}}\mathbf{Q}_{\text{ext}}\mathbf{G}_{\text{ext}}, \tag{42}$$

where d is the per-head channel dimension. Finally, we map back to the spatial domain via $\mathbf{\Psi}_{\text{int}}\mathbf{Y}_{\text{int}}, \mathbf{\Psi}_{\text{ext}}\mathbf{Y}_{\text{ext}}$. Importantly, all transformations in this block are invariant to the arbitrary choice of eigenbasis (i.e. its symmetries) discussed in Section 6.

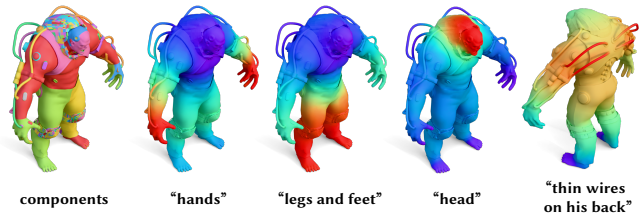

Fig. 18. Steklov-CLIP's semantic saliency maps on a mesh with 80 connected components. Our model is even able to discern thin features (right).

To recapitulate, each network block contains two spectral operations: Steklov heat, which applies a volumetric diffusion-like filter on boundary features, and the Steklov-Galerkin transform above, which learns a data-dependent bilinear interaction between functions represented in the Steklov function spaces. Finally, resulting features from these operations are mixed with an MLP. We summarize the core network block in Algorithm 3.

Training. We train our Steklov-CLIP model for 45,000 iterations on 6×B200 GPUs, using a total batch size of 960. We use AdamW with $\beta_1 = 0.9, \beta_2 = 0.99$ and a cosine annealing learning rate schedule with warmup, taking the initial learning rate as `1e-7` scaling to `6e-4` over 500 steps, then tapering to `3e-4` over 4000 steps. We use weight decay of 0.01, which is not applied to any rank-1 trainable parameters (e.g. layer scales and learned heat times). Our network is trained with `BFloat16` precision and the learned InfoNCE temperature, τ, used in Eq. 35 is capped to 50. Finally, our network uses xyz coordinates, normals, and RGB colors as input.

Evaluation. We evaluate Steklov-CLIP on the canonical zero-shot Objaverse-LVIS classification benchmark. Zero-shot classification is performed through cosine similarity of shape embeddings (as predicted by the network) against text embeddings derived from LVIS category labels. In particular, the category labels are augmented through several text templates, e.g. `a detailed 3D model of a {category}`, whose associated embeddings are averaged to form a final target embedding. LVIS contains 46,207 shapes across 1156 categories. In Table 1, we report top-1 and top-5 zero-shot classification accuracies of our method against representative point cloud-based and multi-view methods. Figure 15 shows a qualitative probe of our model's representations, demonstrating that Steklov-CLIP is able to correctly understand even subtle aspects of input meshes. We further evaluate fine-grained representations learned by these models by performing a dense classification task over all points on the shape. In particular, we use part-level captions from the Partverse [Dong et al. 2025] dataset to perform per-point retrieval over all part captions associated with each shape. Table 2 shows retrieval accuracies in this setting, which are greatly improved by the fine-tuning strategy discussed below.

Fine-grained semantic queries. CLIP-style contrastive pre-training is a proven tool for high-level semantic understanding; however, previous literature has demonstrated its limitations in extracting meaningful fine-grained representations, i.e. ones that can semantically localize parts of a subject [Zhong et al. 2022; Mukhoti et al. 2023; Tschannen et al. 2025]. To enhance the fine-grained capability of our

Table 1. Zero-shot classification scores on Objaverse-LVIS. Shape embeddings are evaluated by nearest-neighbor classification against augmented LVIS category labels. We compare to representative point cloud (PC) and multi-view (MV) methods. † Steklov-CLIP acts on a point-based functional basis (see Sec. 4.4) but uses our mesh-aware Steklov spectra.

Objaverse LVIS Zero-shot Classification				
Method	**Rep**	**Params**	**Top1**	**Top5**
ULIP [2023]	PC	151M	26.8	52.6
OpenShape [2023]	PC	32M	46.8	77.0
Uni3D-base [2024]	PC	88M	51.7	80.8
Uni3D-giant [2024]	PC	1B	55.3	82.9
ViT-Lens [2024]	PC	233M	52.0	79.9
CLIP [2021]	MV	87M	35.7	62.1
Duoduo CLIP [2025]	MV	87M	55.2	83.4
Steklov-CLIP *(ours)*	Mesh†	53M	49.1	76.4

Table 2. We evaluate fine-grained semantic alignment by retrieving part captions using per-point shape embeddings. The `Point` metric classifies each surface point by its nearest part caption embedding. The `Part` metric first pools per-point embeddings over each ground-truth part before retrieval. *FT* denotes our finetuned model, which is evaluated on held-out data.

Partverse Part-Caption Retrieval		
Method	**Part (T1/T5)**	**Point (T1/T5)**
OpenShape [2023]	32.1 / 80.8	23.9 / 75.4
Uni3D-base [2024]	35.8 / 82.3	32.8 / 81.0
Steklov-CLIP	43.4 / 85.7	35.9 / 82.2
Steklov-CLIP *(FT)*	60.1 / 93.0	54.3 / 91.8

Steklov-CLIP model, we finetune it on a small collection of shapes with part-level captions taken from the Partverse dataset [Dong et al. 2025]. We finetune on just 6995 shapes and reserve 1230 for validation. Each shape has a variable number of parts with corresponding captions, we embed the captions using the same OpenCLIP checkpoint as in our pre-training. We finetune using an objective that encourages both part-level and per-point semantic alignment to these part captions—in particular, we use

$$\mathcal{L}_{\text{finetune}} = \mathcal{L}_{\text{part}} + \mathcal{L}_{\text{point}} + \gamma \mathcal{L}_{\text{reg}} \tag{43}$$

where $\mathcal{L}_{\text{part}}$ computes InfoNCE between pooled point embeddings on each part; similarly, $\mathcal{L}_{\text{point}}$ contrasts all point embeddings to each part directly, and $\mathcal{L}_{\text{reg}}$ is a low-weight cosine similarity between part shape/text embeddings, taking $\gamma = 0.1$. We finetune for only 3500 steps using a batch size of 720 and learning rate of `5e-6`.

Even this modest amount of finetuning greatly improves Steklov-CLIP's ability to localize semantic geometric features. Figures 16 and 18 show qualitative results in which Steklov-CLIP's point-wise cosine similarity is visualized w.r.t. a given text query. These saliency maps are filtered through a short-time Steklov heat equation to remove noise—see Figure 19 for raw saliency maps. Our model

is able to isolate semantic parts, and even generalizes to unusual cases—e.g. the two-headed Demogorgon with tentacles.

Remarks. The preceding evaluations suggest that Steklov-CLIP is perhaps more parameter and data efficient as compared to existing methods, given that our model is trained only on ∼50% of the data as cited works. Further, our model's increased performance on fine-grained evaluation (Table 2) likely benefits from: i) not requiring tokenization—unlike the PointBERT transformers used in cited point cloud methods; and b) our use of geometric operators that directly encode local effects. Finally, several methods initialize directly from pretrained weights of internet-scale vision models (i.e. Duoduo CLIP, Uni3D, ViT-Lens), and hence direct comparison should be qualified—this characteristic may also affect the aforementioned models' abilities to localize features, if for example they are too reliant on non-geometric features that are acquired from initialization.

8 Conclusion

We have demonstrated a practical algorithm for volumetric spectral geometry processing using the Dirichlet-to-Neumann operator. The scalability and robustness of our Monte Carlo method unlocks the utility of these geometric constructs, and has allowed us to apply volumetric techniques to large-scale mesh representation learning, which previously was only viable through point cloud-based and multi-view methods.

Discussion & Limitations. Our method is not without limitations—most apparent is that our method employs a spectral approximation to the estimated DtN operators. For many practical applications in geometry processing, spectral approximations are welcomed, though we recognize potential applications in applied sciences that may require higher frequency components in the approximated operators. Further, resolving the *highest*-frequency eigenmodes of such operators may be entirely impractical with Monte Carlo, due to the extreme variations and numerical sensitivity at such scales.

While our CUDA implementation of the estimators in Section 4 is extremely fast, there remain several axes to improve performance. First, as noted by Figure 17, our exterior DtN estimator is bottlenecked by closest-point queries to the Kelvin-inverted surface, which are more computationally expensive than ordinary point-triangle queries. We suspect more optimizations can be made to the acceleration structures employed, as well as the CUDA code in general (i.e. by reducing overhead in the exterior estimator's kernels). Second, the Galerkin bases discussed in Section 4.4 follow from sparse matrices and hence rely on CPU-based eigendecomposition routines. While these bases are effective for our applications, we foresee opportunity to use simpler (and cheaper) options—e.g. Random Fourier Features (RFF) derived directly from vertex coordinates [Rahimi and Recht 2007]. Especially for large meshes, we are primarily bottlenecked by Galerkin basis computation rather than the Monte Carlo estimator itself. Finally, while our estimators are well-defined in open domains (so long as conventional surface normals are determined), its runtime suffers due to WoS samples taking longer to converge and requiring theoretically unbounded steps. We suspect the application domains at hand would benefit from methods that reduce this burden by introducing bias—much to the dismay of Monte Carlo puritans.

We are excited by the prospect of expanding this methodology to more volumetric operators beyond DtN. For instance, even the Poisson kernel itself immediately yields a spectral decomposition of interest [Auchmuty 2017], to which our method can be directly applied. Second, we hope to see more geometric methods built upon the exterior harmonic processes explored in this paper, e.g. for further downstream applications to multi-component geometry.

Acknowledgments

The authors thank Noam Aigerman for his involvement and supervision of foundational work from which the early seeds of the presented ideas were born. We also thank Nicole Ge for her involvement in initial experimentation. Finally, the authors are thankful for fruitful discussions with Nick Sharp, Sina Nabizadeh, Ana Dodik, Derek Liu, Keenan Crane, and Siddhartha Chaudhuri.

References

Valeri I Agoshkov. 1988. Poincaré-Steklov operators and domain decomposition methods in finite dimensional spaces. In *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, Vol. 1. 73.

Mathieu Aubry, Ulrich Schlickewei, and Daniel Cremers. 2011. The wave kernel signature: A quantum mechanical approach to shape analysis. In *2011 IEEE international conference on computer vision workshops (ICCV workshops)*. IEEE, 1626–1633.

Giles Auchmuty. 2017. The SVD of the Poisson kernel. *Journal of Fourier Analysis and Applications* 23, 6 (2017), 1517–1536.

S. Axler, P. Bourdon, and W. Ramey. 2006. *Harmonic Function Theory*. Springer New York. https://books.google.fr/books?id=MYHbBwAAQBAJ

A. Beurling and J. Deny. 1958. Espaces de dirichlet. *Acta Mathematica* 99, 1 (1958), 203–224. doi:10.1007/BF02392426

Ilia Binder and Mark Braverman. 2012. The rate of convergence of the Walk on Spheres Algorithm. *Geometric and Functional Analysis* 22, 3 (2012), 558–587. doi:10.1007/s00039-012-0161-z

Robert Bridson. 2007. Fast Poisson disk sampling in arbitrary dimensions. *SIGGRAPH sketches* 10, 1 (2007), 1.

Michael M Bronstein and Iasonas Kokkinos. 2010. Scale-invariant heat kernel signatures for non-rigid shape recognition. In *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 1704–1711.

Shuhao Cao. 2021. Choose a transformer: Fourier or galerkin. *Advances in neural information processing systems* 34 (2021), 24924–24940.

Zhen-Qing Chen and Masatoshi Fukushima. 2011. *Symmetric Markov Processes, Time Change, and Boundary Theory (LMS-35)*. Princeton University Press. http://www.jstor.org/stable/j.ctt7s6w6

Etienne Corman, Justin Solomon, Mirela Ben-Chen, Leonidas Guibas, and Maks Ovsjanikov. 2017. Functional characterization of intrinsic and extrinsic geometry. *ACM Transactions on Graphics (TOG)* 36, 2 (2017), 1–17.

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. 2023. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 13142–13153.

Mathieu Desbrun, Mark Meyer, and Pierre Alliez. 2002. Intrinsic parameterizations of surface meshes. In *Computer graphics forum*, Vol. 21. Wiley Online Library, 209–218.

Ana Dodik, Vincent Sitzmann, Justin Solomon, and Oded Stein. 2025. Robust biharmonic skinning using geometric fields. *ACM Transactions on Graphics* 45, 2 (2025), 1–18.

Shaocong Dong, Lihe Ding, Xiao Chen, Yaokun Li, Yuxin Wang, Yucheng Wang, Qi Wang, Jaehyeok Kim, Chenjian Gao, Zhanpeng Huang, et al. 2025. From one to more: Contextual part latents for 3d generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 8230–8240.

Masatoshi Fukushima, Yoichi Oshima, and Masayoshi Takeda. 1994. *Dirichlet Forms and Symmetric Markov Processes*. De Gruyter, Berlin, New York. doi:doi:10.1515/9783110889741

Alexander Gao, Maurice Chu, Mubbasir Kapadia, Ming C Lin, and Hsueh-Ti Derek Liu. 2024. An intrinsic vector heat network. *arXiv preprint arXiv:2406.09648* (2024).

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.

Qi-Xing Huang, Martin Wicke, Bart Adams, and Leonidas Guibas. 2009. Shape decomposition using modal analysis. In *Computer Graphics Forum*, Vol. 28. Wiley Online

Library, 407–416.
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*. 1–12.
Gabriel Ilharco, Mitchell Wortsman, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, et al. 2021. Openclip. *Zenodo* (2021).
Alec Jacobson, Ilya Baran, Jovan Popovic, and Olga Sorkine. 2011. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.* 30, 4 (2011), 78.
Alec Jacobson, Ladislav Kavan, and Olga Sorkine-Hornung. 2013. Robust inside-outside segmentation using generalized winding numbers. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–12.
Clément Jambon, Mohammad Sina Nabizadeh, and Mina Konaković Luković. 2026. Walk on Decomposed Subdomains: A Hybrid Monte Carlo–Deterministic Solver for Elliptic PDEs. *ACM Trans. Graph.* 45, 4, Article 132 (July 2026), 22 pages. doi:10.1145/3811340
Shizuo Kakutani. 1944. 143. Two-dimensional Brownian Motion and Harmonic Functions. *Proceedings of the Imperial Academy* 20, 10 (1944), 706–714. doi:10.3792/pia/1195572706
Karl Karlovič Sabel'fel'd and Nikolai A Simonov. 1994. *Random walks on boundaries for solving PDES*. VSP.
Han-Hung Lee, Yiming Zhang, and Angel Chang. 2025. Duoduo CLIP: Efficient 3D understanding with multi-view images. In *International Conference on Learning Representations*, Vol. 2025. 48070–48091.
Weixian Lei, Yixiao Ge, Kun Yi, Jianfeng Zhang, Difei Gao, Dylan Sun, Yuying Ge, Ying Shan, and Mike Zheng Shou. 2024. Vit-lens: Towards omni-modal representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 26647–26657.
Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. 2023. Least squares conformal maps for automatic texture atlas generation. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*. 193–202.
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. doi:10.1145/3641519.3657395
Or Litany, Tal Remez, Emanuele Rodola, Alex Bronstein, and Michael Bronstein. 2017. Deep functional maps: Structured prediction for dense shape correspondence. In *Proceedings of the IEEE international conference on computer vision*. 5659–5667.
Hsueh-Ti Derek Liu, Alec Jacobson, and Keenan Crane. 2017. A Dirac operator for extrinsic shape analysis. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 139–149.
Minghua Liu, Ruoxi Shi, Kaiming Kuang, Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su. 2023. Openshape: Scaling up 3d shape representation towards open-world understanding. *Advances in neural information processing systems* 36 (2023), 44860–44879.
Arman Maesumi, Tanish Makadia, Thibault Groueix, Vladimir Kim, Daniel Ritchie, and Noam Aigerman. 2025. PoissonNet: A Local-Global Approach for Learning on Surfaces. *ACM Transactions on Graphics (TOG)* 44, 6 (2025), 1–16.
Bailey Miller, Rohan Sawhney, Keenan Crane, and Ioannis Gkioulekas. 2023. Boundary value caching for walk on spheres. *arXiv preprint arXiv:2302.11825* (2023).
Bailey Miller, Rohan Sawhney, Keenan Crane, and Ioannis Gkioulekas. 2024a. Differential walk on spheres. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–18.
Bailey Miller, Rohan Sawhney, Keenan Crane, and Ioannis Gkioulekas. 2024b. Walkin'robin: Walk on stars with robin boundary conditions. *ACM Transactions on Graphics* 43, 4 (2024).
Jishnu Mukhoti, Tsung-Yu Lin, Omid Poursaeed, Rui Wang, Ashish Shah, Philip HS Torr, and Ser-Nam Lim. 2023. Open vocabulary semantic segmentation with patch aligned contrastive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 19413–19423.
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
Mohammad Sina Nabizadeh, Ravi Ramamoorthi, and Albert Chern. 2021. Kelvin transformations for simulations on infinite domains. *ACM Trans. Graph.* 40, 4, Article 97 (July 2021), 15 pages. doi:10.1145/3450626.3459809
Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).
Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. 2012. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (ToG)* 31, 4 (2012), 1–11.
Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PmLR, 8748–8763.
Ali Rahimi and Benjamin Recht. 2007. Random features for large-scale kernel machines. *Advances in neural information processing systems* 20 (2007).
Martin Reuter. 2010. Hierarchical shape segmentation and registration via topological features of Laplace-Beltrami eigenfunctions. *International Journal of Computer Vision* 89, 2 (2010), 287–308.
Rohan Sawhney and Keenan Crane. 2017. Boundary first flattening. *ACM Transactions on Graphics (ToG)* 37, 1 (2017), 1–14.
Rohan Sawhney and Keenan Crane. 2020. Monte Carlo geometry processing: A grid-free approach to PDE-based methods on volumetric domains. *ACM Transactions on Graphics* 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. *arXiv preprint arXiv:2302.11815* (2023).
Rohan Sawhney, Dario Seyb, Wojciech Jarosz, and Keenan Crane. 2022. Grid-free Monte Carlo for PDEs with spatially varying coefficients. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–17.
Nicholas Sharp, Souhaib Attaiki, Keenan Crane, and Maks Ovsjanikov. 2022. Diffusionnet: Discretization agnostic learning on surfaces. *ACM Transactions on Graphics (ToG)* 41, 3 (2022), 1–16.
Nicholas Sharp and Keenan Crane. 2020. A laplacian for nonmanifold triangle meshes. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 69–80.
Dmitriy Smirnov and Justin Solomon. 2021. HodgeNet: Learning spectral geometry on triangle meshes. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–11.
Olga Sorkine, Marc Alexa, et al. 2007. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, Vol. 4. 109–116.
I. Stakgold and M.J. Holst. 2011. *Green's Functions and Boundary Value Problems*. Wiley. https://books.google.com/books?id=8OeoISCW6qUC
Ryusuke Sugimoto, Terry Chen, Yiti Jiang, Christopher Batty, and Toshiya Hachisuka. 2023. A practical walk-on-boundary method for boundary value problems. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–16.
Jian Sun, Maks Ovsjanikov, and Leonidas Guibas. 2009. A concise and provably informative multi-scale signature based on heat diffusion. In *Computer graphics forum*, Vol. 28. Wiley Online Library, 1383–1392.
Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. 2025. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786* (2025).
Ingo Wald. 2026. cuBQL: A CUDA "BVH Build-and-Query" Library. https://github.com/NVIDIA/cuBQL.
Yu Wang, Mirela Ben-Chen, Iosif Polterovich, and Justin Solomon. 2018. Steklov Spectral Geometry for Extrinsic Shape Analysis. *ACM Trans. Graph.* 38, 1, Article 7 (Dec. 2018), 21 pages. doi:10.1145/3152156
Yu Wang and Justin Solomon. 2019. Intrinsic and extrinsic operators for shape analysis. In *Handbook of numerical analysis*. Vol. 20. Elsevier, 41–115.
Ruben Wiersma, Ahmad Nasikun, Elmar Eisemann, and Klaus Hildebrandt. 2022. Deltaconv: anisotropic operators for geometric deep learning on point clouds. *ACM Transactions on Graphics (ToG)* 41, 4 (2022), 1–10.
Le Xue, Mingfei Gao, Chen Xing, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. 2023. Ulip: Learning a unified representation of language, images, and point clouds for 3d understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 1179–1189.
Zi Ye, Olga Diamanti, Chengcheng Tang, Leonidas Guibas, and Tim Hoffmann. 2018. A unified discrete framework for intrinsic and extrinsic Dirac operators for geometry processing. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 93–106.
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
Yiwu Zhong, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luowei Zhou, Xiyang Dai, Lu Yuan, Yin Li, et al. 2022. Regionclip: Region-based language-image pretraining. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 16793–16803.
Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. 2024. Uni3d: Exploring unified 3d representation at scale. In *International Conference on Learning Representations*, Vol. 2024. 46766–46782.
Qingnan Zhou and Alec Jacobson. 2016. Thingi10k: A dataset of 10,000 3d-printing models. *arXiv preprint arXiv:1605.04797* (2016).

A Practical input assumptions

The derivations in Section 4 are written with respect to a smooth boundary $\partial\Omega$ of a volumetric domain Ω. In practice, however, our implementation is applied to meshes from minimally-curated shape collections, where surfaces may be open, inconsistently oriented, multi-component, or locally degenerate. This section describes the practical convention used to turn such inputs into two-sided surfaces on which the interior and exterior estimators can be defined.

For watertight consistently oriented meshes, the convention agrees with the usual notion of inside and outside. For partially open surfaces or those with inconsistent normals, we instead use the generalized winding number (GWN) of Jacobson et al. [2013] to define a canonical outward direction. This construction provides a consistent assignment of sidedness for interior and exterior WoS samples. To summarize:

(1) **Do we require closed surfaces?** No, we do not require input meshes to be watertight. Normals are used to determine the direction in which interior and exterior tangent balls are drawn, from which walks are sampled. When the input orientation is inconsistent, we repair normals using generalized winding numbers, orienting each face according to the local inside-outside convention. In extreme cases (e.g. a plane) the sides of a surface become geometrically symmetric; in this setting, our interior and exterior operators should be interpreted as the two indistinguishable halves of a "two-sided" operator, rather than as distinct operators associated with inside and outside.

(2) **What about self-intersections?** The largest inscribed tangent ball is not well-defined arbitrarily close to an intersection point; in the limit, its radius approaches zero. In practice, we do not require the mesh to be intersection-free. Samples near these singularities are discarded, while samples away from the intersection point are unchanged. The algebraic properties of our estimator (i.e. its PSD guarantee) are unaffected, though the extent of the discarded region may affect bias.

(3) **What if the largest tangent ball is unbounded?** We follow the heuristic of Yu et al. [2024] by clamping the maximum diameter for tangent balls to the shortest side length of the mesh's axis-aligned bounding box. For the exterior estimator, we follow the same heuristic, though the bounding box is taken w.r.t. the Kelvin-inverted surface.

(4) **How is nested geometry treated?** Nested geometry introduces a modeling ambiguity that is not resolved by the surface geometry alone. For example, if a closed component lies entirely inside another closed component, one may interpret the inner component as an internal obstacle, a cavity boundary, a separate solid object, or an artifact to be ignored, depending on the intended application. Our implementation adopts the convention that the entire input surface is absorbing: any triangle may serve as a terminating boundary for WoS samples. Thus, nested components are treated as additional boundary components of the sampled domain, rather than being discarded. This convention is appropriate when the entire surface is intended to participate in the boundary process, but it is ultimately a modeling choice; applications with different semantics may wish to remove nested components, or assign them different boundary behavior.

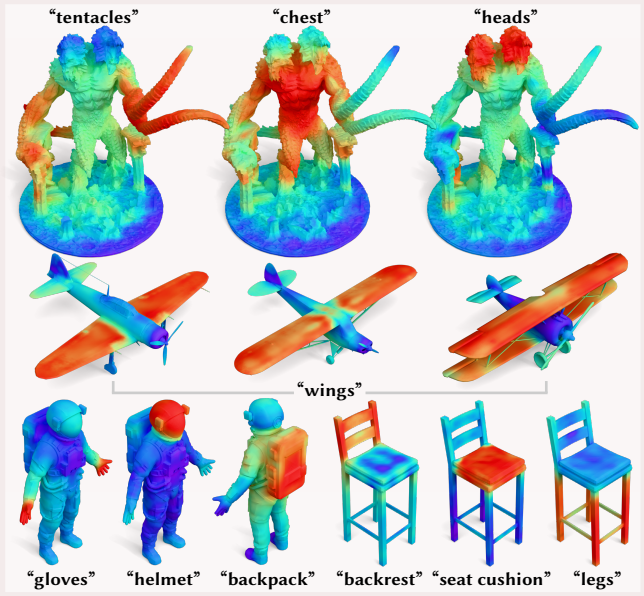

Fig. 19. Unfiltered saliency maps from Steklov-CLIP, corresponding to the *filtered* saliency maps in Figure 16.

B Point-based k-NN Interpolator

Our point-based estimators employ k-NN interpolators when evaluating Galerkin basis functions on the boundary—i.e., to compute $b_s = \phi(s)$ and $b_t = \phi(t)$ for a given boundary function ϕ (analogous to the barycentric interpolation used in Algorithm 2). To reduce spurious interactions between boundary points, we use a bilateral weighting on the k nearest neighbors around the query point.

In particular, given a query point q, we find its k nearest point cloud samples $\mathcal{N}_k(q)$. The value of a boundary function ϕ at q is approximated by a weighted average over these neighbors,

$$\phi(q) \approx \sum_{i \in \mathcal{N}_k(q)} w_i(q)\, \phi(x_i), \tag{44}$$

where the bilateral weights, w_i, depend on both spatial distance and normal alignment,

$$\begin{aligned} \widetilde{w}_i(q) &= \exp\left(-\frac{\|q - x_i\|^2}{2\sigma^2}\right) \exp\left(-\frac{1 - \langle n_q, n_i\rangle}{\tau_n}\right), \\ w_i(q) &= \frac{\widetilde{w}_i(q)}{\sum_{j \in \mathcal{N}_k(q)} \widetilde{w}_j(q)}. \end{aligned} \tag{45}$$

Here, x_i and n_i are the position and normal of the i-th point, and n_q is the normal at the query point. The parameter σ determines spatial falloff, and the parameter τ_n determines normal falloff. In our implementation, we use $\sigma = 0.025$, $\tau_n = 0.8$, and $k = 8$ neighbors.

C Derivations for Interior and Exterior DtN Estimators

This supplemental section fills in the derivations for the interior and exterior Beurling-Deny bilinear forms used in Section 4, together with the closed-form identities that underlie our samplers.

Algorithm 3. Steklov Network Block

Input: Boundary features $\mathbf{f} \in \mathbb{R}^{N\times C}$; mass matrix $\mathbf{M}$; Steklov eigenbases of size k, $(\mathbf{\Psi}_{\text{int}}, \lambda_{\text{int}})$, $(\mathbf{\Psi}_{\text{ext}}, \lambda_{\text{ext}})$
Output: Updated boundary features $\mathbf{f}_{\text{out}} \in \mathbb{R}^{N\times C}$

$\mathbf{f}_{\text{in}} \leftarrow \mathbf{f}$

▷ *Apply interior/exterior spectral heat filters with learned times* $\mathbf{t}$
$\mathbf{h} \leftarrow \textsc{SteklovHeat}(\mathbf{f}, \mathbf{\Psi}_{\text{int}}, \lambda_{\text{int}}, \mathbf{\Psi}_{\text{ext}}, \lambda_{\text{ext}}, \mathbf{M}, \mathbf{t})$
$\mathbf{f} \leftarrow \mathbf{f} + \gamma_{\text{heat}} \odot \mathbf{h}$

▷ *Steklov-Galerkin Attention:*
$\mathbf{u} \leftarrow \mathbf{f}\,\mathbf{W}_{\text{in}}$
for $s \in \{\text{int}, \text{ext}\}$ **do**
  ▷ *Projection into Steklov spectrum*
  $\mathbf{Z}_s \leftarrow \mathbf{\Psi}_s^{\mathsf{T}}\mathbf{M}\mathbf{u}$
  ▷ *Project to spectral queries, keys, and values*
  $\mathbf{Q}_s \leftarrow \mathbf{Z}_s\mathbf{W}_s^{Q}, \quad \mathbf{K}_s \leftarrow \mathbf{Z}_s\mathbf{W}_s^{K}, \quad \mathbf{V}_s \leftarrow \mathbf{Z}_s\mathbf{W}_s^{V}$
  ▷ *Modulate using eigenvalue-dependent spectral filters*
  $\mathbf{Q}_s \leftarrow \textsc{Modulate}(\mathbf{Q}_s, \boldsymbol{\lambda}_s, \boldsymbol{\eta}_s^{Q})$
  $\mathbf{K}_s \leftarrow \textsc{Modulate}(\mathbf{K}_s, \boldsymbol{\lambda}_s, \boldsymbol{\eta}_s^{K})$
  $\mathbf{V}_s \leftarrow \textsc{Modulate}(\mathbf{V}_s, \boldsymbol{\lambda}_s, \boldsymbol{\eta}_s^{V})$
  ▷ *Assemble bilinear operator from Steklov functions*
  $\mathbf{G}_s \leftarrow \frac{1}{k}\mathbf{K}_s^{\mathsf{T}}\mathbf{V}_s$
end for
▷ *Learned interior/exterior mixing (per-head)*
$\widetilde{\mathbf{G}}_{\text{int}} \leftarrow \alpha_{\text{ii}}\mathbf{G}_{\text{int}} + \alpha_{\text{ie}}\mathbf{G}_{\text{ext}}$
$\widetilde{\mathbf{G}}_{\text{ext}} \leftarrow \alpha_{\text{ei}}\mathbf{G}_{\text{int}} + \alpha_{\text{ee}}\mathbf{G}_{\text{ext}}$
▷ *Apply operators to queries*
$\mathbf{Y}_{\text{int}} \leftarrow \frac{1}{\sqrt{d}}\,\mathbf{Q}_{\text{int}}\widetilde{\mathbf{G}}_{\text{int}}, \quad \mathbf{Y}_{\text{ext}} \leftarrow \frac{1}{\sqrt{d}}\,\mathbf{Q}_{\text{ext}}\widetilde{\mathbf{G}}_{\text{ext}}$
▷ *Project back to spatial domain and apply linear layer*
$\mathbf{a} \leftarrow (\mathbf{\Psi}_{\text{int}}\mathbf{Y}_{\text{int}} + \mathbf{\Psi}_{\text{ext}}\mathbf{Y}_{\text{ext}})\,\mathbf{W}_{\text{out}}$
$\mathbf{f} \leftarrow \mathbf{f} + \gamma_{\text{attn}} \odot \mathbf{a}$

▷ *Mix updated/original features*
$\mathbf{m} \leftarrow \text{MLP}([\mathbf{f}_{\text{in}}, \mathbf{f}])$
$\mathbf{f}_{\text{out}} \leftarrow \mathbf{f} + \gamma_{\text{mlp}} \odot \mathbf{m}$
return $\mathbf{f}_{\text{out}}$

function $\textsc{SteklovHeat}(\mathbf{f}, \mathbf{\Psi}_{\text{int}}, \lambda_{\text{int}}, \mathbf{\Psi}_{\text{ext}}, \lambda_{\text{ext}}, \mathbf{M}, \mathbf{t})$
  ▷ *Evolve features using interior/exterior Steklov heat per-channel*
  $\mathbf{f}_{\text{int}}, \mathbf{f}_{\text{ext}} \leftarrow \textsc{SplitChannels}(\mathbf{f})$ *Each has $C/2$ channels*
  $\mathbf{t}_{\text{int}}, \mathbf{t}_{\text{ext}} \leftarrow \textsc{SplitChannels}(\mathbf{t})$ *Learned heat times per channel*
  $\mathbf{h}_{\text{int}}^{c} \leftarrow \mathbf{\Psi}_{\text{int}} \begin{bmatrix} e^{-\lambda_{\text{int}}^{0} t_{\text{int}}^{c}} \\ \dots \\ e^{-\lambda_{\text{int}}^{k} t_{\text{int}}^{c}} \end{bmatrix} \odot (\mathbf{\Psi}_{\text{int}}^{\mathsf{T}}\mathbf{M}\,\mathbf{f}_{\text{int}}^{c})$
  $\mathbf{h}_{\text{ext}}^{c} \leftarrow \mathbf{\Psi}_{\text{ext}} \begin{bmatrix} e^{-\lambda_{\text{ext}}^{0} t_{\text{ext}}^{c}} \\ \dots \\ e^{-\lambda_{\text{ext}}^{k} t_{\text{ext}}^{c}} \end{bmatrix} \odot (\mathbf{\Psi}_{\text{ext}}^{\mathsf{T}}\mathbf{M}\,\mathbf{f}_{\text{ext}}^{c})$
  return $\textsc{Concat}(\mathbf{h}_{\text{int}}, \mathbf{h}_{\text{ext}})$
end function

function $\textsc{Modulate}(\mathbf{X}_s, \lambda_s, \boldsymbol{\eta}_s^{\bullet})$
  ▷ *Modulate spectral function with multi-scale heat kernel*
  $\boldsymbol{\tau} \leftarrow \{\tau_1, \dots, \tau_L\}$ *Fixed log-spaced heat times*
  $\boldsymbol{\omega}_s^{\bullet} \leftarrow \text{softmax}_{\ell}(\boldsymbol{\eta}_s^{\bullet})$ *Learned convex weights over heat scales*
  $\left[\mathbf{D}_s^{\bullet}\right]_{kk} \leftarrow \sum_{\ell=1}^{L} \omega_{s,\ell}^{\bullet} \exp(-\tau_\ell \lambda_k^s)$
  return $\mathbf{D}_s^{\bullet}\mathbf{X}_s$
end function

Throughout this section, we translate from the conventions of Chen and Fukushima [2011], who formulate the DtN operator through the Dirichlet form $(\frac{1}{2}\mathbf{D}, H^1(\Omega))$ with generator $\frac{1}{2}\Delta$ (here, $\mathbf{D}[f,g] := \int_\Omega \nabla f \cdot \nabla g \, dV$ is the Dirichlet integral). Our convention (Eq. 12) uses the unscaled Dirichlet form $(\mathbf{D}, H^1(\Omega))$ generated by Δ, so equations from this source pick up a factor of 2 when brought into our notation. Recall that our outward normals n_s at boundary points $s \in \partial\Omega$ always point *out* of Ω and *into* Ω_{ext}.

C.1 Beurling-Deny Formula

Beurling-Deny Decomposition of DtN. Chen and Fukushima [2011, Eq. 5.8.4] write the Dirichlet energy of a function $f \in H^{\frac{1}{2}}(\partial\Omega)$ as

$$\mathcal{E}[f,f] = \frac{1}{2}\iint_{\partial\Omega\times\partial\Omega} \big(f(s)-f(t)\big)^2 J^{\Omega}(s,t)\, ds\, dt. \tag{46}$$

Applying the *Polarization Identity* to Eq. 46 gives the bilinear form

$$\mathcal{E}[f,g] = \frac{1}{2}\iint_{\partial\Omega\times\partial\Omega} \big(f(s)-f(t)\big)\big(g(s)-g(t)\big)\, J^{\Omega}(s,t)\, ds\, dt, \tag{47}$$

which is the starting point for deriving our interior estimator. A similar application of the *Polarization Identity* to [Chen and Fukushima 2011, Eq. 5.8.9] provides the starting point for deriving our exterior estimator.

C.2 Jump Kernel

Jump Kernel of a Ball. After establishing the jump kernel as the inward normal derivative of the Poisson kernel [Chen and Fukushima 2011, Eq. 5.8.2], we can derive a closed-form expression for the jump measure between a point $s \in \partial\Omega$ and another point z on the boundary of a ball $B_r(c) \subset \Omega$ that is tangent to the boundary at s. The ball has radius $r > 0$, center $s - rn_s$, and Poisson kernel

$$P^{B_r(c)}(x \to z) = \frac{1}{4\pi r}\,\frac{r^2 - |x-c|^2}{|x-z|^3}, \tag{48}$$

which vanishes when x lies on $\partial B_r(c)$. Therefore, we have

$$\begin{aligned}
J^{B_r(c)}(s,z) &= -\partial_{n_s} P^{B_r(c)}(s \to z) \\
&= \lim_{\epsilon\to 0} \frac{P^{B_r(c)}(s-\epsilon n_s \to z) - P^{B_r(c)}(s \to z)}{\epsilon} \\
&= \lim_{\epsilon\to 0} \frac{P^{B_r(c)}(s-\epsilon n_s \to z)}{\epsilon} \\
&= \lim_{\epsilon\to 0} \frac{1}{4\pi r\epsilon}\,\frac{r^2 - |(s-\epsilon n_s)-c|^2}{|(s-\epsilon n_s)-z|^3} \\
&= \lim_{\epsilon\to 0} \frac{1}{4\pi r\epsilon}\,\frac{r^2 - (r-\epsilon)^2}{|(s-\epsilon n_s)-z|^3} \\
&= \lim_{\epsilon\to 0} \frac{1}{4\pi r\epsilon}\,\frac{r^2 - (r^2 - 2r\epsilon + \epsilon^2)}{|(s-\epsilon n_s)-z|^3} \\
&= \lim_{\epsilon\to 0} \frac{1}{4\pi r}\,\frac{2r-\epsilon}{|(s-\epsilon n_s)-z|^3} \\
&= \frac{1}{4\pi r}\,\frac{2r}{|s-z|^3} \\
&= \frac{1}{2\pi|s-z|^3}.
\end{aligned} \tag{49}$$

This closed-form ball jump kernel is the analytical building block that lets us factor WoS-style walks out of the general jump kernel.

Jump Kernel of Ω. The ball jump kernel $J^{B_r(c)}$ is known in closed-form, but the jump kernel J^{Ω} for a general domain is not. To maximize sampling efficiency, we would like to leverage the analytically known jump measure between two ball surface points in order to factor out the portion of a jump on $\partial\Omega$ that can be handled exactly. By writing the jump kernel in terms of the Poisson kernel (Eq. 18) and using the *Strong Markov Property* (Eq. 21), we obtain

$$\begin{aligned} J^{\Omega}(s,t) &= -\partial_{n_s} P^{\Omega}(s \to t) \\ &= -\partial_{n_s} \int_{\partial B_r(c)} P^{B_r(c)}(s \to z)\, P^{\Omega}(z \to t)\, dz \\ &= \int_{\partial B_r(c)} -\partial_{n_s} P^{B_r(c)}(s \to z)\, P^{\Omega}(z \to t)\, dz \\ &= \int_{\partial B_r(c)} J^{B_r(c)}(s,z)\, P^{\Omega}(z \to t)\, dz. \end{aligned} \tag{50}$$

The resulting factorization reveals that the singular, differential portion of J^{Ω} is fully captured by the known ball jump kernel.

C.3 Kelvin Transform

The *Kelvin transform* is a conformal mapping $\kappa : \mathbb{R}^3 \cup \{\infty\} \to \mathbb{R}^3 \cup \{\infty\}$ sending $x \mapsto x/|x|^2$ (with $0 \mapsto \infty$ and $\infty \mapsto 0$). Assuming $0 \in \Omega$ (which can always be arranged by translation), we define

$$\Omega^{\star}_{\text{ext}} := \kappa(\Omega_{\text{ext}} \cup \{\infty\}) \tag{51}$$

as the inverted exterior. Under the Kelvin transform, $\Omega^{\star}_{\text{ext}}$ is a *bounded* domain in $\mathbb{R}^3$ that includes the origin (as the image of ∞). We denote the image of x under the Kelvin transform as $x^{\star} := \kappa(x)$. It is important to recognize that κ is its own inverse, so that $\kappa(x^{\star}) = x$.

Jacobian of the Kelvin Transform. The ij-th entry of the Kelvin transform's Jacobian is

$$\mathbf{J}_{\kappa}(x)_{ij} = \frac{\partial x_i^{\star}}{\partial x_j} = \frac{\partial \left(x_i/|x|^2\right)}{\partial x_j}.$$

Applying the *Quotient Rule*, we have that $\mathbf{J}_{\kappa}(x)_{ij}$ is equal to

$$\frac{|x|^2 \frac{\partial x_i}{\partial x_j} - x_i \frac{\partial \sum x_i^2}{\partial x_j}}{|x|^4} = \frac{\delta_{ij}|x|^2 - 2x_i x_j}{|x|^4} = \frac{1}{|x|^2}\left(\delta_{ij} - \frac{2x_i x_j}{|x|^2}\right).$$

In matrix form, the Jacobian simplifies to

$$\mathbf{J}_{\kappa}(x) = \frac{1}{|x|^2}\left(\mathbf{I} - \frac{2xx^{\mathsf{T}}}{|x|^2}\right) = \frac{1}{|x|^2}\mathbf{H}_{\hat{x}}, \tag{52}$$

where $\mathbf{H}_{\hat{x}}$ is the *Householder reflection* about the direction $\hat{x} = x/|x|$.

Boundary Scaling. Based on its Jacobian, the linear scale factor of κ is $\ell(x) = 1/|x|^2$. Surface area elements on a 2D manifold scale by $\ell(x)^2$, which means

$$d\sigma(s^{\star}) = \frac{1}{|s|^4} d\sigma(s) \quad \text{and} \quad d\sigma(s) = \frac{1}{|s^{\star}|^4} d\sigma(s^{\star}). \tag{53}$$

In general, this means we can transform a surface integral on the inverted boundary $\partial\Omega^{\star}$ into an integral on the primal boundary $\partial\Omega$ by introducing the appropriate density-correcting weight $|s|^{-4}$.

Normal Inversion. Applying the Kelvin transform's Jacobian (Eq. 52) to n_s reflects the normal about the radial axis $\hat{s}$ and introduces a local scaling term,

$$\mathbf{J}_{\kappa}(s)\, n_s = \frac{1}{|s|^2}\mathbf{H}_{\hat{s}}\, n_s = \frac{1}{|s|^2}\,(-n_{s^{\star}}) = -|s^{\star}|^2 n_{s^{\star}}, \tag{54}$$

where $n_{s^{\star}}$ denotes the unit normal—pointing out of $\Omega^{\star}_{\text{ext}}$ and into $\Omega^{\star}$—defined at points $s^{\star} \in \partial\Omega^{\star}$ on the inverted boundary.

Kelvin Transform of a Function. Let u be a function defined on $\Omega_{\text{ext}} \subset \mathbb{R}^3 \setminus \{0\}$. Then, the function $\kappa[u]$ defined on $\Omega^{\star}_{\text{ext}} \setminus \{0\}$ by

$$\kappa[u](x^{\star}) := |x|\, u(x) \tag{55}$$

is the *Kelvin transform of u*. In particular, if u is harmonic on Ω_{ext} and satisfies $u(x) \to 0$ as $x \to \infty$, then the transformation $\kappa[u]$ is harmonic on $\Omega^{\star}_{\text{ext}}$ [Axler et al. 2006].

C.4 Exterior Poisson Kernel

Kelvin Transform of the Poisson Kernel. We wish to relate the Poisson kernels of the two harmonic functions given in Eq. 55 (namely u and $\kappa[u]$). Suppose $u = \mathcal{H}^{\Omega_{\text{ext}}}[g]$ for some boundary data $g : \partial\Omega \to \mathbb{R}$. By Eq. 4, we obtain the Poisson integral

$$u(x) = \int_{\partial\Omega} g(s)\, P^{\Omega_{\text{ext}}}(x \to s)\, ds \tag{56}$$

for $x \in \Omega_{\text{ext}}$. Then, $\kappa[u]$ solves the *bounded* Dirichlet problem on $\Omega^{\star}_{\text{ext}}$ with boundary conditions $\kappa[u](s^{\star}) = |s|\, g(s)$ for $s^{\star} \in \partial\Omega^{\star}$. Its Poisson integral takes the form

$$\kappa[u](x^{\star}) = \int_{\partial\Omega^{\star}} |s|\, g(s)\, P^{\Omega^{\star}_{\text{ext}}}(x^{\star} \to s^{\star})\, ds^{\star}, \tag{57}$$

this time for $x^{\star} \in \Omega^{\star}_{\text{ext}}$. We can represent this as a boundary integral on the primal surface $\partial\Omega$ by introducing a $|s|^{-4}$ scale factor (Eq. 53),

$$\kappa[u](x^{\star}) = \int_{\partial\Omega} |s|^{-3}\, g(s)\, P^{\Omega^{\star}_{\text{ext}}}(x^{\star} \to s^{\star})\, ds. \tag{58}$$

By definition (Eq. 55), $\kappa[u](x^{\star}) = |x|\, u(x)$, which implies that

$$\int_{\partial\Omega} |s|^{-3}\, g(s)\, P^{\Omega^{\star}_{\text{ext}}}(x^{\star} \to s^{\star})\, ds = |x| \int_{\partial\Omega} g(s)\, P^{\Omega_{\text{ext}}}(x \to s)\, ds.$$

Since $|x|^{-1} = |x^{\star}|$, the $|x|$ factor can be moved to the left-hand side,

$$\int_{\partial\Omega} g(s)\, |x^{\star}|\, |s^{\star}|^3\, P^{\Omega^{\star}_{\text{ext}}}(x^{\star} \to s^{\star})\, ds = \int_{\partial\Omega} g(s)\, P^{\Omega_{\text{ext}}}(x \to s)\, ds.$$

Here, g is an arbitrary function. Because the two integrals must agree pointwise for any choice of g, we reach the identity

$$P^{\Omega_{\text{ext}}}(x \to s) = |x^{\star}|\, |s^{\star}|^3\, P^{\Omega^{\star}_{\text{ext}}}(x^{\star} \to s^{\star}) \tag{59}$$

relating the Poisson kernel of the exterior domain Ω_{ext} with that of the inverted domain $\Omega^{\star}_{\text{ext}}$. This identity underpins our exterior DtN estimator formulation, which exploits the boundedness of $\Omega^{\star}_{\text{ext}}$.

C.5 Exterior Jump Kernel

Like the interior case (Eq. 18), the exterior jump kernel is the normal derivative of the exterior Poisson kernel,

$$J^{\Omega_{\text{ext}}}(s,t) = \partial_{n_s} P^{\Omega_{\text{ext}}}(s \to t). \tag{60}$$

Substituting the exterior Poisson kernel relation in Eq. 59,

$$\begin{aligned} J^{\Omega_{\text{ext}}}(s,t) &= \partial_{n_s}\left[|s^\star|\,|t^\star|^3\,P^{\Omega^\star_{\text{ext}}}(s^\star \to t^\star)\right] \\ &= \nabla_s\left[|s^\star|\,|t^\star|^3\,P^{\Omega^\star_{\text{ext}}}(s^\star \to t^\star)\right]\cdot n_s \\ &= |t^\star|^3\left[\,P^{\Omega^\star_{\text{ext}}}(s^\star \to t^\star)\,\nabla_s\,|s^\star| + |s^\star|\,\nabla_s\,P^{\Omega^\star_{\text{ext}}}(s^\star \to t^\star)\right]\cdot n_s. \end{aligned}$$

The first term vanishes because $P^{\Omega^\star_{\text{ext}}}(s^\star \to t^\star) = 0$, leaving

$$J^{\Omega_{\text{ext}}}(s,t) = |s^\star|\,|t^\star|^3\,\nabla_s\,P^{\Omega^\star_{\text{ext}}}(s^\star \to t^\star)\cdot n_s. \tag{61}$$

By the chain rule,

$$\begin{aligned} \nabla_s\,P^{\Omega^\star_{\text{ext}}}(s^\star \to t^\star)\cdot n_s &= \mathbf{J}_\kappa(s)^\mathsf{T}\,\nabla_{s^\star}\,P^{\Omega^\star_{\text{ext}}}(s^\star \to t^\star)\cdot n_s \\ &= \nabla_{s^\star}\,P^{\Omega^\star_{\text{ext}}}(s^\star \to t^\star)\cdot\mathbf{J}_\kappa(s)\,n_s \\ &= \nabla_{s^\star}\,P^{\Omega^\star_{\text{ext}}}(s^\star \to t^\star)\cdot\left(-|s^\star|^2\,n_{s^\star}\right) \\ &= -|s^\star|^2\,\nabla_{s^\star}\,P^{\Omega^\star_{\text{ext}}}(s^\star \to t^\star)\cdot n_{s^\star} \\ &= -|s^\star|^2\,\partial_{n_{s^\star}}\,P^{\Omega^\star_{\text{ext}}}(s^\star \to t^\star) \\ &= |s^\star|^2\,J^{\Omega^\star_{\text{ext}}}(s^\star,t^\star), \end{aligned} \tag{62}$$

where the third equality applies the image of the surface normal under Kelvin inversion (Eq. 54) and the final equality applies the interior jump kernel definition (Eq. 18) on the bounded domain $\Omega^\star_{\text{ext}}$. Assembling, we attain

$$J^{\Omega_{\text{ext}}}(s,t) = |s^\star|^3\,|t^\star|^3\,J^{\Omega^\star_{\text{ext}}}(s^\star,t^\star). \tag{63}$$

The exterior jump kernel on the unbounded domain Ω_{ext} is—up to a pair of distortion-correcting scale factors—the interior style jump kernel on the bounded inverted domain $\Omega^\star_{\text{ext}}$. Notably, $J^{\Omega^\star_{\text{ext}}}$ admits the very same tangent-ball decomposition which we used to reduce variance in the interior case (Eq. 24).

C.6 Killing Measure

Escape Probability. The escape probability $q : \Omega_{\text{ext}} \to [0,1]$ on exterior points is defined as

$$q(x) := 1 - \mathcal{H}^{\Omega_{\text{ext}}}[\mathbf{1}](x), \tag{64}$$

which is harmonic on the unbounded exterior domain Ω_{ext}, vanishes on $\partial\Omega$, and approaches 1 as $|x| \to \infty$ [Chen and Fukushima 2011, Eq. 5.8.6]. We claim that $q(x)$ is equivalent to

$$\tilde{q}(x) := -4\pi\,|x^\star|\,G^{\Omega^\star_{\text{ext}}}(x^\star,0), \tag{65}$$

which is defined on the unbounded exterior domain $\Omega_{\text{ext}} \cup \{\infty\}$. Here, $G^{\Omega^\star_{\text{ext}}}$ is the *Green's function* of the inverted domain, given by

$$G^{\Omega^\star_{\text{ext}}}(x^\star,0) := -\frac{1}{4\pi|x^\star|} + u(x^\star,0), \tag{66}$$

for some u that is harmonic on $\Omega^\star_{\text{ext}}$ [Stakgold and Holst 2011, Exercise 8.3.2]. After substituting Eq. 66 into Eq. 65, we obtain

$$\tilde{q}(x) = 1 - 4\pi\,|x^\star|\,u(x^\star,0). \tag{67}$$

There are two things to notice here:

(1) **Boundary conditions.** $G^{\Omega^\star_{\text{ext}}}(x^\star,0)$ vanishes on $\partial\Omega^\star$, which means that $\tilde{q}(x)$ also vanishes on $\partial\Omega$—matching the behavior of $q(x)$ on the boundary.

(2) **Limit at infinity.** Because $u(x^\star,0)$ is harmonic on $\Omega^\star_{\text{ext}}$, which includes 0, it is bounded on a neighborhood of $x^\star = 0$. Hence,

$$\lim_{x^\star\to 0} 4\pi\,|x^\star|\,u(x^\star,0) = 0. \tag{68}$$

As $|x| \to \infty$, we have $|x^\star| \to 0$. By Eq. 67 and Eq. 68, we get

$$\lim_{x\to\infty}\tilde{q}(x) = 1 - 0 = 1, \tag{69}$$

which agrees with $q(x)$ at infinity.

By the uniqueness of exterior Dirichlet problem solutions, we must conclude that $q(x) = \tilde{q}(x)$. In other words, escape probability in the exterior domain Ω_{ext} is equivalently written in terms of the Green's function of the Laplacian in the inverted domain $\Omega^\star_{\text{ext}}$, such that

$$q(x) = -4\pi\,|x^\star|\,G^{\Omega^\star_{\text{ext}}}(x^\star,0). \tag{70}$$

This re-characterization will become vital for deriving an expression for the killing measure $K^{\Omega_{\text{ext}}}$ that can actually be sampled.

Sampling Killing Measure. The starting point for our killing measure estimator comes from Chen and Fukushima [2011, Eq. 5.8.6],

$$K^{\Omega_{\text{ext}}}(s) = \partial_{n_s}\,q(s). \tag{71}$$

Substituting in our work from the preceeding section (Eq. 70),

$$\begin{aligned} K^{\Omega_{\text{ext}}}(s) &= \partial_{n_s}\left(-4\pi\,|s^\star|\,G^{\Omega^\star_{\text{ext}}}(s^\star,0)\right) \\ &= \nabla_s\left(-4\pi\,|s^\star|\,G^{\Omega^\star_{\text{ext}}}(s^\star,0)\right)\cdot n_s \\ &= -4\pi\left[G^{\Omega^\star_{\text{ext}}}(s^\star,0)\,\nabla_s\,|s^\star| + |s^\star|\,\nabla_s\,G^{\Omega^\star_{\text{ext}}}(s^\star,0)\right]\cdot n_s. \end{aligned}$$

The first term vanishes because $G^{\Omega^\star_{\text{ext}}}(s^\star,0) = 0$ when $s^\star \in \partial\Omega^\star$ ($G^{\Omega^\star_{\text{ext}}}$ satisfies homogeneous Dirichlet boundary conditions). Then,

$$K^{\Omega_{\text{ext}}}(s) = -4\pi\,|s^\star|\,\nabla_s\,G^{\Omega^\star_{\text{ext}}}(s^\star,0)\cdot n_s. \tag{72}$$

Using the chain rule,

$$\begin{aligned} \nabla_s\,G^{\Omega^\star_{\text{ext}}}(s^\star,0)\cdot n_s &= \mathbf{J}_\kappa(s)^\mathsf{T}\,\nabla_{s^\star}\,G^{\Omega^\star_{\text{ext}}}(s^\star,0)\cdot n_s \\ &= \nabla_{s^\star}\,G^{\Omega^\star_{\text{ext}}}(s^\star,0)\cdot\mathbf{J}_\kappa(s)\,n_s \\ &= \nabla_{s^\star}\,G^{\Omega^\star_{\text{ext}}}(s^\star,0)\cdot\left(-|s^\star|^2\,n_{s^\star}\right) \\ &= -|s^\star|^2\,\nabla_{s^\star}\,G^{\Omega^\star_{\text{ext}}}(s^\star,0)\cdot n_{s^\star} \\ &= -|s^\star|^2\,\partial_{n_{s^\star}}\,G^{\Omega^\star_{\text{ext}}}(s^\star,0) \\ &= -|s^\star|^2\,\partial_{n_{s^\star}}\,G^{\Omega^\star_{\text{ext}}}(0,\,s^\star) \\ &= -|s^\star|^2\,P^{\Omega^\star_{\text{ext}}}(0 \to s^\star), \end{aligned} \tag{73}$$

where the third equality follows from the Kelvin transform of surface normals (Eq. 54), the sixth equality holds because the Dirichlet Green's function is symmetric in its two arguments, and the last equality identifies the normal derivative of the Green's function as the Poisson kernel (Eq. 7). Substituting this back into Eq. 72,

$$K^{\Omega_{\text{ext}}}(s) = 4\pi\,|s^\star|^3\,P^{\Omega^\star_{\text{ext}}}(0 \to s^\star), \tag{74}$$

which remarkably implies that we can sample the killing measure by starting random walks from the inversion center (i.e. $\kappa(\infty)$, which is defined to be the origin).

C.7 Exterior DtN Estimator

Let $\{\phi_k\}_{k=1}^K$ be our chosen boundary basis, let $b(s) \in \mathbb{R}^K$ be a pointwise evaluation of the basis at $s \in \partial\Omega$, and let $d(s,t) := b(s) - b(t) \in \mathbb{R}^K$ denote the difference of basis evaluations at two different boundary points. Using the Beurling Deny form given in Eq. 27, the reduced *exterior* Steklov matrix $\mathrm{S}_{\mathrm{ext}} \in \mathbb{R}^{K\times K}$, with entries equal to $(\mathrm{S}_{\mathrm{ext}})_{ij} = \mathcal{E}[\phi_i, \phi_j]$, is expressed as

$$\begin{aligned}\mathrm{S}_{\mathrm{ext}} = \frac{1}{2} \iint\limits_{\partial\Omega\times\partial\Omega} d(s,t)\, d(s,t)^{\mathsf{T}} J^{\Omega_{\mathrm{ext}}}(s,t)\, ds\, dt \\ + \int_{\partial\Omega} b(s)\, b(s)^{\mathsf{T}} K^{\Omega_{\mathrm{ext}}}(s)\, ds.\end{aligned} \tag{75}$$

Our goal is to trace the formulation of the *interior* PSD-by-construction estimator of the DtN operator (Section 4.2), but using the Kelvin-transformed proxies for the exterior jump kernel (Eq. 63) and killing measure (Eq. 74) instead. Mirroring the tangent-ball decomposition in Section C.2, the exterior domain jump kernel factors as

$$J^{\Omega_{\mathrm{ext}}}(s,t) = |s^\star|^3\, |t^\star|^3 \int_{\partial B_r(c^\star)} J^{B_r(c^\star)}(s^\star, z^\star)\, P^{\Omega^\star_{\mathrm{ext}}}(z^\star \to t^\star)\, dz^\star.$$

After substituting this quantity into the jump term of Eq. 75, and substituting Eq. 74 into its killing term, we can follow the same rearrangement that precedes Eq. 25, yielding

$$\begin{aligned}\mathrm{S}_{\mathrm{ext}} = \frac{1}{2} \iint\limits_{\partial\Omega\times\partial B_r(c^\star)} |s^\star|^3\, J^{B_r(c^\star)}(s^\star, z^\star) \\ \cdot \left(\int_{\partial\Omega} d(s,t)\, d(s,t)^{\mathsf{T}}\, |t^\star|^3\, P^{\Omega^\star_{\mathrm{ext}}}(z^\star \to t^\star)\, dt \right) dz^\star\, ds \\ + \left(\int_{\partial\Omega} b(t)\, b(t)^{\mathsf{T}}\, 4\pi\, |t^\star|^3\, P^{\Omega^\star_{\mathrm{ext}}}(0 \to t^\star)\, dt \right).\end{aligned} \tag{76}$$

The variable of integration of the two parenthesized boundary integrals is $t \in \partial\Omega$ on the primal surface. However, both integrals contain the Poisson kernel of the Kelvin-transformed domain $\Omega^\star_{\mathrm{ext}}$. For consistency, we can switch the variable of integration to $t^\star \in \partial\Omega^\star$ on the inverted boundary using the relation given in Eq. 53,

$$\begin{aligned}\mathrm{S}_{\mathrm{ext}} = \frac{1}{2} \iint\limits_{\partial\Omega\times\partial B_r(c^\star)} |s^\star|^3\, J^{B_r(c^\star)}(s^\star, z^\star) \\ \cdot \left(\int_{\partial\Omega^\star} d(s,t)\, d(s,t)^{\mathsf{T}}\, \frac{1}{|t^\star|}\, P^{\Omega^\star_{\mathrm{ext}}}(z^\star \to t^\star)\, dt^\star \right) dz^\star\, ds \\ + \left(\int_{\partial\Omega^\star} b(t)\, b(t)^{\mathsf{T}}\, \frac{4\pi}{|t^\star|}\, P^{\Omega^\star_{\mathrm{ext}}}(0 \to t^\star)\, dt^\star \right).\end{aligned} \tag{77}$$

Now, both parentheticals are Poisson integrals (Eq. 4) and can be converted into expectations (Eq. 5),

$$\begin{aligned}\mathrm{S}_{\mathrm{ext}} = \frac{1}{2} \iint\limits_{\partial\Omega\times\partial B_r(c^\star)} \mathbb{E}_{t^\star}\left[d(s,t)\, d(s,t)^{\mathsf{T}} \frac{1}{|t^\star|} \right] |s^\star|^3\, J^{B_r(c^\star)}(s^\star, z^\star)\, dz^\star\, ds \\ + \mathbb{E}_{t_0^\star}\left[b(t_0)\, b(t_0)^{\mathsf{T}}\, \frac{4\pi}{|t_0^\star|} \right],\end{aligned} \tag{78}$$

where the terminal WoS exit points $t^\star \sim \omega_{z^\star}^{\Omega^\star_{\mathrm{ext}}}$ and $t_0^\star \sim \omega_0^{\Omega^\star_{\mathrm{ext}}}$ are sampled in the inverted domain. At last, by sampling surface points $s \sim U(\partial\Omega)$ on the primal boundary and $z^\star \sim U(\partial B_r(c^\star))$ on the inverted tangent ball surface, we can write an *exterior* estimator that operates in the Kelvin-transformed domain,

$$\begin{aligned}\mathrm{S}_{\mathrm{ext}} = \mathop{\mathbb{E}}_{s,\, z^\star,\, t^\star}\left[\frac{|\partial\Omega|\, |\partial B_r(c^\star)|\, |s^\star|^3}{2\, |t^\star|}\, d(s,t)\, d(s,t)^{\mathsf{T}}\, J^{B_r(c^\star)}(s^\star, z^\star) \right] \\ + \mathop{\mathbb{E}}_{t_0^\star}\left[\frac{4\pi}{|t_0^\star|}\, b(t_0)\, b(t_0)^{\mathsf{T}} \right].\end{aligned} \tag{79}$$

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009