Smooth Geometric Representations for Online Adaptation in Robotic Manipulation

THIS IS A TEMPORARY TITLE PAGE
It will be replaced for the final print by a version
provided by the registrar's office.

Thèse n. 1234 2026
présentée le 10 août 2026
à la Faculté des sciences et techniques de l'ingénieur
Laboratoire de traitement des signaux 4
Programme doctoral en génie électrique
École polytechnique fédérale de Lausanne

pour l'obtention du grade de Docteur ès Sciences
par

Cem Bilaloglu

acceptée sur proposition du jury :

Prof A. Cavallaro, président du jury
Prof P. Frossard, Dr S. Calinon, directeurs de thèse
Prof T. Murphey, rapporteur
Prof Stefan Ivić, rapporteur
Prof Stefana Parascho, rapporteuse

Lausanne, EPFL, 2026

To my family

Acknowledgements

Navigating a PhD is a remarkable journey full of highs and lows. I am profoundly grateful to Ceren, my family, friends, and colleagues who made me feel loved and supported throughout.

A special thank you goes to Sylvain for his guidance and unwavering support; this work would not have been possible without him. The same goes for Tobias, with whom I have worked and shared so many great moments since day one. I would also like to especially thank my collaborators and friends Luca, Ziyi and Max for our discussions that directly shaped my work.

I would like to thank my lifelong friends, Utku, Efe, Egehan, Alper, as well as the new friends met during these years: Hakan, Mert, Sai, Amir, Ante, Boyang, Guillaume, James, Jeremy, Jiacheng, Karlo, Luka, Martin, Saray, Teng, Vivek, Xuemin, Yan, Yiming. They all have been a source of joy, support, and inspiration throughout my PhD journey in their unique ways.

Words cannot fully express my gratitude to my mother, father, and sisters. There was never a single moment where I did not feel their love and support. I am grateful to my grandmother for her boundless love and care while raising me, and I owe a very special thank you to my sister, Gizem, who is my greatest inspiration and supporter in my academic journey.

Finally, my deepest thanks go to Ceren. She has been my everything throughout this journey, bringing warmth and joy to every step, and filling it with unforgettable memories.

Sion, August 10, 2026 Cem Bilaloglu

Abstract

Enabling robots to operate autonomously in previously unseen environments is a fundamental goal in modern robotics. To achieve this, robots must explore their environments and adapt known skills to new settings.

However, achieving this autonomy is hindered by several critical challenges: (i) immense continuous variation of human environments requires online adaptation, (ii) representations built online rely on partial and uncertain sensing, and (iii) online adaptation requires high computational efficiency. Compounding these issues is the fact that robot manipulation data fundamentally lives on curved spaces, encompassing complex geometries of multi-link robots, arbitrary object surfaces and rigid body motions.

This thesis proposes that smoothness and geometry can induce an interdependent structure for efficient and robust representations built online. By bridging intrinsic geometry with spatial smoothness through diffusion processes and the spectrum of differential operators, we establish mathematical regularizers that can cope with environmental variations, sensor noise and missing data. This interdependent structure allows representations to be computed in real-time directly on raw, incomplete sensor inputs such as point clouds without requiring complete reconstructions.

We demonstrated the efficacy of these smooth geometric representations across a diverse range of tasks essential for a robot operating in previously unknown environments, including online whole-body exploration, tactile exploration on curved surfaces, and task adaptation to object and environmental variations. Through real-world experimental validation, we demonstrated robustness to sensor noise, missing data and unmodeled dynamics while maintaining the computational efficiency essential for online planning and control.

Keywords: Robot Manipulation, Tactile Robotics, Object-centric Representations, Ergodic Control, Diffusion Processes, Spectral Geometry, Differential Geometry

Résumé

Permettre aux robots d'opérer de manière autonome dans des environnements totalement inédits constitue un objectif fondamental de la robotique moderne. Pour y parvenir, les robots doivent être capables d'explorer leur milieu et d'adapter les compétences acquises à de nouveaux contextes.

Cependant, l'accès à une telle autonomie est freiné par plusieurs défis majeurs : (i) la variabilité continue et infinie des environnements humains exige une adaptation en ligne, (ii) les représentations construites en temps réel reposent sur une perception partielle et incertaine, et (iii) l'adaptation en ligne requiert une efficacité algorithmique extrêmement élevée. Ces difficultés sont accentuées par le fait que les données de manipulation robotique évoluent fondamentalement dans des variétés riemanniennes, englobant la géométrie complexe des systèmes poly-articulés, les surfaces d'objets arbitraires et les déplacements rigides.

Cette thèse soutient que la régularité spatiale et la géométrie peuvent induire une structure interdépendante permettant de construire en ligne des représentations à la fois efficaces et robustes. En reliant la géométrie intrinsèque à la régularité spatiale par le biais de processus de diffusion et du spectre d'opérateurs différentiels, nous établissons des régulariseurs mathématiques capables de surmonter les variations environnementales, le bruit des capteurs et les données manquantes. Cette structure interdépendante permet de calculer des représentations en temps réel directement à partir de données sensorielles brutes et incomplètes (telles que des nuages de points), sans nécessiter de reconstruction géométrique complète.

Nous démontrons l'efficacité de ces représentations géométriques régulières à travers une grande variété de tâches essentielles à l'évolution d'un robot en environnement inconnu. Celles-ci incluent l'exploration en ligne exploitant le corps entier du robot, l'exploration tactile sur des surfaces courbes, ainsi que l'adaptation des compétences aux variations des objets et de l'environnement. Les validations expérimentales sur des plateformes réelles démontrent la robustesse de notre approche face au bruit des capteurs, aux pertes de données et aux dynamiques non modélisées, tout en préservant l'efficacité algorithmique indispensable à la planification et au contrôle en temps réel.

Contents

1 Introduction

Unlike controlled factory settings, human environments present continuous and unbounded variation. Consequently, it is impossible to pre-program a robot for every scenario it may encounter. Instead, as emphasized in the recent AI for robotics roadmap [15], achieving true autonomy in previously unseen environments requires online adaptation: robots must be capable of actively exploring their surroundings and transferring known manipulation skills to novel settings. This thesis posits that this transition requires not simply an increase in data, unlike the recent trend seen in foundation models [41, 8], but the development of interpretable and modular representations embedded with the correct inductive biases, such as smoothness and intrinsic geometry.

The shift from factory floors to human environments necessitates a fundamental change in hardware and control paradigms. Industrial robots utilize highly repeatable sensing and actuation systems to execute deterministic, pre-planned trajectories. Although effective for repeatable tasks and environments, these rigid systems (i) lack the flexibility required to adapt to environmental variation and (ii) are particularly ill-suited for tactile tasks such as cleaning or peeling, where the physical interactions are prohibitively complex to model. In contrast, robots operating in human-centric environments and tasks must transition from a reliance on rigid position repeatability and precise models towards representations that are constructed and updated on-the-fly from imperfect sensing. However, unlike industrial manipulators leveraging industrial-grade instrumentation, robots for human environments must utilize onboard hardware such as consumer-grade depth cameras and force-torque sensors. These sensors are inherently subject to measurement noise, occlusions, and epistemic uncertainty, yielding noisy and partial data. Consequently, this thesis proposes the use of spatial smoothness and intrinsic geometry as fundamental regularizers to construct robust representations for perception, control and learning that are built and updated online.

Compounding these sensory challenges is that tactile manipulation tasks naturally occur on curved surfaces (non-Euclidean manifolds). The surface of a robot manipulator itself consists of multiple links and joints with complex geometries. Existing approaches model the manipulator as a single point [1] whereas it can be leveraged as a whole-body contact sensor with massive footprint for exploring the surrounding environment. Similarly, it is common to use reduced models for objects such as keypoints [66, 33, 32, 31], convex hulls [34, 81] or geometric primitives [51, 113, 35]. Although these abstractions are highly effective for coarse tasks such as pick-and-place and obstacle avoidance, tactile manipulation demands representations that explicitly and precisely account for the curved geometry of target objects. However, human-centric environments present a severe challenge: everyday items (such as fruits or kitchen utensils) exhibit unbounded intra- and inter-category shape variations. The precise, watertight surface meshes required by traditional planning algorithms [43] are fundamentally unavailable for arbitrary, real-world objects and modern vision-based learning methods [23] do not explicitly account for the geometry. To bypass these limitations, we relax the requirement for either perfect or reduced models and operate directly on partial, noisy point clouds collected online. By leveraging diffusion processes [1], also known as heat equation, as a regularizer, we transform this raw, partial sensory data into smooth geometric representations. This framework ultimately enables tactile exploration and task transfer across object shape and environmental variations.

1.1 Core Contributions & Thesis Outline

Centered on the challenges of a manipulator adapting online to previously unseen environments, we investigate the interpretable structures of geometry and smoothness that can be systematically reused across tasks, objects, and environmental variations. Ultimately, this thesis aims to show that:

> **By bridging data, geometry, and spatial smoothness, diffusion processes provide an inductive bias for online adaptation in robotic manipulation.**

We leverage this inductive bias in whole-body exploration (Chapter 3), exploration and estimation on curved surfaces (Chapter 4), task adaptation across environmental variations (Chapter 5), and object shapes (Chapter 6). The core contributions of this thesis are organized within the subsequent chapters as follows:

- **Whole-Body Spatial Exploration** (Chapter 3): We extended ergodic control, which is conventionally restricted to idealized point agents, to the complex geometry of a robot manipulator surface. By solving the diffusion PDE, we formulated a locally smooth field that coordinates a collection of kinematically constrained virtual agents, transforming the robot's entire kinematic chain into an extended tactile sensor.
- **Tactile Ergodic Control on Curved Surfaces** (Chapter 4): We extended online ergodic control to arbitrary curved surfaces represented as raw point clouds. By unifying the

[1] We use the partial differential equation (PDE) or stochastic differential equation (SDE) governing spreading of particles or information from high density to low density, which differ from *diffusion models* used as generative models in machine learning [38] and policy learning [23]. We discuss the similarities and differences in Section 1.2.3

exploration controller and a heat-kernel-based Gaussian Process estimation module under the same structural foundation of diffusion, we enabled online mapping of physical properties (such as viscoelasticity) on curved objects.

- **Adaptive Ergodic Imitation** (Chapter 5): We presented a framework that bridges imitation and exploration for addressing the brittleness of standard imitation learning under environmental shifts. By leveraging geometry-aware anisotropic diffusion modulated by estimated stagnation, the controller smoothly morphs from rigid imitation to goal-oriented exploration when faced with environmental mismatches.

- **Shape-Invariant Skill Transfer** (Chapter 6): We introduced a modular, geometry-aware representation for object-centric manipulation tasks. By combining discrete surface diffusion on point clouds with continuous workspace diffusion, our method constructs a smooth field of object-centric local reference frames. This decouples the task description from object shapes, allowing object-centric tasks to seamlessly transfer to previously unseen object shapes at runtime.

1.2 Related Work

1.2.1 Diffuse Control Targets and Objectives using Ergodic Control

In repeatable industrial settings, manipulation tasks are traditionally formulated as point-reaching or trajectory-tracking problems that direct an end-effector to discrete spatial coordinates. Under this paradigm, the control objective is minimized in terms of the distance between the end-effector and the target point. In contrast, *ergodic control* introduced by Mathew and Mezic [68] generalizes this objective by treating spatial distributions as the target. The optimization goal shifts from reaching a static coordinate to achieving ergodicity, meaning that the time-averaged statistics of the robot's trajectory match the spatial statistics of the target distribution. Consequently, an ergodic controller dictates that the robot spends its time budget in regions proportionally to their local density, spending more time on high-utility zones compared to low-utility areas. From this viewpoint, traditional point-reaching is simply a limiting case in which the target distribution becomes highly concentrated (mathematically, a Dirac delta) at a single location. At the other extreme, one can consider a uniform spatial distribution as the target, which would yield a coverage path planner (CPP) that exhaustively explores the environment. Realistic scenarios, however, occupy a middle ground between these two extremes. A purely uniform target leading to maximizing entropy [12] is often unnecessary because human-centric environments exhibit structural patterns where previous experiences can act as informative priors. Conversely, a non-smooth Dirac target leading to information-maximizing [71] is often unattainable or brittle due to inherent mismatches in environments and uncertainties in sensing, modeling, and actuation.

In addition to using diffuse targets, ergodic control fundamentally relies on smooth discrepancy measures which compares the target distribution against robot's spatial coverage (trajectory empirical distribution). Spectral Multiscale Coverage (SMC) [68], the foundational algorithm in the field, achieves this by representing both distributions in Fourier basis and minimizing their difference using a negative Sobolev space norm. Defining the metric in the negative Sobolev space captures the smoothness of a function by incorporating spatial derivatives in addition to its magnitude, thereby prioritizing large-scale, smooth coverage behavior over high-frequency details. Alternatively, the Heat-Equation Driven Area Coverage (HEDAC) [44] algorithm employs the stationary heat (diffusion) equation to evolve the mismatch between the target and the actual coverage into a smooth potential field. Conceptually, these approaches are deeply intertwined: from a historical perspective, Fourier basis were originally developed to orthogonalize the diffusion PDE to a system of ordinary differential equations (ODEs), and from a spectral perspective, the diffusion equation is a multiscale low-pass filter just like the SMC. Similarly, recent advancements such as the kernel ergodic metric [94] reformulate the L^2 distance between distributions by relaxing the Dirac delta using smooth kernels such as the Gaussian kernel. Notably, Gaussian kernel is equivalent to the heat kernel (fundamental solution of the diffusion equation) in Euclidean spaces and heat kernel provides a geometry-aware and multiscale extension of Gaussian kernels in Riemannian manifolds [16, 6, 7]. Ultimately, whether through Sobolev norms, diffusion, or smoothing kernels, formulating the ergodic objective via smooth discrepancy measures ensures that the control policy remains robust to non-smooth targets and discrete, noisy coverage data, a critical advantage over brittle L^2 distance-based formulations.

1.2.2 Diffusion-based Intrinsic Representations for Curved Objects

Most existing control, planning, and learning frameworks, including standard ergodic control [68], assume flat domains. On the curved surfaces of real-world objects (Riemannian manifolds), these assumptions break down: the shortest path between two points (the geodesic) is rarely a straight line, vectors cannot be trivially compared across different points without accounting for curvature through parallel transport, and classical Fourier analysis are inapplicable.

Compounding this geometric complexity is the challenge of generalization across varying configurations of the same object, as well as the immense intra- and inter-category shape variations of real-world objects. Current robotic foundation models [41, 8] frequently treat these variations as a data scarcity problem, relying on brute-force data augmentation or synthetic demonstration generation [104] to statistically force the network to learn task-relevant symmetries, such as equivariances and invariances. Alternatively, geometric deep learning [17] approaches focus on embedding pose equivariance directly into network architectures [90, 106, 112, 110]. However, their success in generalizing across structural shape variations remains a critical open problem. We argue for a more modular paradigm: utilizing spatial representations that possess the desired invariances by design, thereby providing a shared, mathematically guaranteed foundation for control, trajectory optimization, and learning.

We propose that diffusion processes governed by the Laplacian operator [2] provide the principled mathematical framework required to realize such representations. At first sight, diffusion processes might seem as a very specific dynamical process but, remarkably, it contains enough information to fully reconstruct all the calculus and geometry of the space [49]. Derived solely from the Riemannian metric, the Laplacian is inherently invariant to isometric transformations encapsulating most of the interesting transformations in robotics such as rigid body motion, articulation, and bending without stretching. This stands in sharp contrast to extrinsic models common in visuomotor policies [23, 109], which fluctuate based on how an object is embedded in 3D space. Furthermore, the Laplacian serves as the fundamental bridge between spatial geometry and spectral analysis, yielding an orthogonal eigenbasis that generalizes classical Fourier series to curved domains and non-rectangular boundaries. This connection allows us to re-derive spectral tools, such as frequency-based prioritization in ergodic control, directly on the manifold.

The spectral perspective enables a natural, multiscale transition from local features to global symmetries governed by the diffusion time parameter. At small scales, high-frequency eigenvalues resolve sharp edges and local features. At larger scales, low-frequency eigenvalues reveal global topology, such as distinguishing a cylinder from a sphere. The fundamental solution of the diffusion PDE, known as the heat kernel, can be constructed from the spectrum of the Laplacian. This construction results in a mathematically robust, geometry-aware similarity measure for Gaussian Process (GP) estimation [16, 6, 7]. Diffusion and heat kernel inherently bypasses the mathematical vulnerabilities of standard geodesic metrics [30]. Backed by Varadhan's Lemma [99], which asymptotically links short-time diffusion to geodesic distance, the diffusion equation integrates energy propagation across all possible paths between points. This makes diffusion-based methods for computing geodesic distances, parallel transport and signed distances [24, 85, 29] robust against the sensor noise, structural defects, and partial occlusions inherent to robotic point clouds. However, an additional challenge specific to robotics is that manipulation skills are rarely constrained strictly to an object's surface; actions like scooping or peeling originate in free space before transitioning into contact and penetration. Classical PDE solvers require full volumetric discretization of the surrounding 3D workspace [29], which is undesirable for a robot moving continuously in space. Although grid-free Monte Carlo alternatives like the Walk on Spheres (WoS) algorithm offer efficient, discretization-free solutions [79, 73], they are currently restricted to flat Euclidean domains. This thesis bridges this gap, by combining discrete Laplacian-based diffusion on point clouds with continous WoS-based workspace diffusion.

1.2.3 Diffusion in Generative Models and Policy Learning

Recently, the term *diffusion* has also gained immense popularity in the machine learning and robotics literature through the lens of generative modeling and policy learning. Diffusion

[2] The extension of Laplacian operator to Riemannian manifolds is known as the Laplace-Beltrami operator, but for conciseness, we will use Laplacian for both.

models refer to a class of generative models which uses forward diffusion process (stochastic version of the PDE we use) to generate data for training reverse diffusion process known as denoising [38]. The learned denoising diffusion process can be used for sampling from the data distribution. Approaches such as Diffusion Policy [23] utilize diffusion models to learn robust perception-to-action mappings from sensory inputs, effectively representing complex, multi-modal action distributions. This generative framework has been rapidly extended to incorporate 3D representations [109] and reactive, contact-rich behaviors using tactile and force feedback [39, 103].

Although mathematically connected, these data-driven approaches have fundamentally different goals from the methods proposed in this thesis. This thesis leverages the diffusion PDE in the physical and spatial domain as a deterministic geometric regularizer whereas diffusion models are class of generative models that requires extensive data and offline training. Still, insights regarding success of diffusion models also inspired some of the design decisions and methods developed in this thesis.

2 Background

In this chapter, our goal is to show how diffusion processes bridges seemingly unrelated concepts such as smoothness, multi-scale intrinsic geometry and ergodicity while underpinning the mathematical and algorithmic foundations of the methods presented in this thesis. We start by introducing Riemannian geometry which is essential for understanding the representation of curved surfaces, and the properties of differential operators on manifolds. Next, we show how diffusion processes can encode the spatial smoothness of a field and be used for computing geodesics and parallel transport. Finally, we introduce ergodic control for spatial exploration and coverage from diffusion processes perspective.

2.1 Mathematical Notation

We denote scalars by lower case letters x, constants by upper case letters X, vectors by boldface lower case letters $\boldsymbol{x}$, matrices by boldface uppercase letters $\boldsymbol{X}$. Manifolds, tangent spaces and sets are designated by calligraphic letters $\mathcal{X}$.

2.2 Riemannian Geometry

To systematically analyze the geometry of arbitrary object surfaces and multi-link robotic systems, we utilize differential geometry. A smooth manifold[1] $\mathcal{M}$ of dimension d is a topological space that locally resembles a d-dimensional Euclidean space $\mathbb{R}^d$. For a point $\boldsymbol{x} \in \mathcal{M}$, the tangent space $\mathcal{T}_{\boldsymbol{x}}\mathcal{M}$ forms a vector space consisting of all tangent vectors to curves passing through $\boldsymbol{x}$.

A *Riemannian manifold* $(\mathcal{M}, g)$ is a smooth manifold equipped with a smoothly varying inner product $g_x \colon \mathcal{T}_{\boldsymbol{x}}\mathcal{M} \times \mathcal{T}_{\boldsymbol{x}}\mathcal{M} \to \mathbb{R}$ on each tangent space, known as the *Riemannian metric*. The metric tensor g generalizes Euclidean geometric concepts such as lengths and angles to curved

[1] The manifold assumption can be relaxed when working with complex or defective point clouds; for a broader theoretical and algorithmic discussion, see [84, 49]

domains. The distance between two points $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{M}$ is defined by the infimum of the length of all continuously differentiable curves connecting them and curves that locally minimize this distance are called *geodesics*. The Riemannian metric also allows us to define the *Levi-Civita connection* ∇, which provides a way to differentiate vector fields along curves on the manifold, and the *parallel transport* of vectors along curves, which preserves the inner product defined by the metric.

2.2.1 Differential Operators on Riemannian Manifolds

Given a smooth scalar function $u : \mathcal{M} \to \mathbb{R}$ on a Riemannian manifold, its gradient at a point $\boldsymbol{x}$, denoted $\nabla_{\mathcal{M}} u(\boldsymbol{x})$, is defined as the unique tangent vector on $\mathcal{T}_{\boldsymbol{x}}\mathcal{M}$ that satisfies the relation

$$g_{\boldsymbol{x}}(\nabla_{\mathcal{M}} u, \boldsymbol{v}) = Du(\boldsymbol{x})[\boldsymbol{v}] \tag{2.1}$$

for any tangent vector $\boldsymbol{v} \in \mathcal{T}_{\boldsymbol{x}}\mathcal{M}$. Intuitively, this condition dictates that the manifold gradient is the unique vector that, when the inner product is computed with any directional tangent vector $\boldsymbol{v}$ via the local Riemannian metric $g_{\boldsymbol{x}}$, yields the exact directional derivative of the function u along that direction $Du(\boldsymbol{x})[\boldsymbol{v}]$.

Conversely, the *divergence* $\mathrm{div}_{\mathcal{M}}(\boldsymbol{v})$ measures the net spatial flux of a vector field $\boldsymbol{v}$ out of a localized region on the manifold

$$\mathrm{div}_{\mathcal{M}}(\boldsymbol{v}) = \mathrm{tr}(\nabla \boldsymbol{v}) \tag{2.2}$$

where we use the Levi-Civita connection ∇ to coordinate-freely differentiate vector fields by tracking how those gradient directions change due to the curvature of the manifold.

The fundamental operator utilized throughout this thesis is the *Laplacian* $\Delta_{\mathcal{M}}$, defined as the divergence of the gradient, or equivalently, the trace of the Hessian tensor dictated by the Levi-Civita connection:

$$\Delta_{\mathcal{M}} u = \mathrm{div}_{\mathcal{M}}(\nabla_{\mathcal{M}} u) = \mathrm{tr}(\nabla^2 u) \tag{2.3}$$

Notably in Euclidean space, the Laplacian reduces to the familiar form $\Delta u = \sum_{i=1}^{d} \frac{\partial^2 u}{\partial \boldsymbol{x}_i^2}$, but on curved manifolds, it intrinsically incorporates the geometry through the metric tensor and connection.

Determined only by the metric g, the Laplacian is inherently invariant to isometric transformations encompassing rigid-body transformations, articulations, and bending without stretching. Furthermore, the operator is negative semi-definite, yielding a discrete spectral decomposition characterized by a sequence of real, non-positive eigenvalues λ_i and corresponding orthogonal eigenfunctions ϕ_i satisfying the Helmholtz equation:

$$\Delta_{\mathcal{M}} \phi_i = \lambda_i \phi_i \tag{2.4}$$

These eigenfunctions provide a generalized Fourier basis directly on the manifold, acting as the structural foundation for both spectral filtering and the spatial diffusion processes used throughout this work.

2.3 Smooth Fields via Diffusion Processes

Consider a scalar field $u : \Omega \to \mathbb{R}$. We can measure the smoothness of the field using the Dirichlet energy functional $E[u]$ penalizing large gradients

$$E[u] = \int_{\Omega} \|\nabla u\|^2 d\boldsymbol{x} = \int_{\Omega} (\nabla u \cdot \nabla u) d\boldsymbol{x}. \tag{2.5}$$

Accordingly, finding the smoothest field over the domain subject to fixed values, known as Dirichlet boundary conditions, is formulated as the following variational optimization problem:

$$\min_{u \in \mathscr{A}} E[u], \tag{2.6}$$

constrained to the admissible set

$$\mathscr{A} = \left\{ u \in H^1(\Omega) : u|_{\partial\Omega} = g \right\}, \tag{2.7}$$

where $H^1(\Omega)$ is the space of functions with square-integrable derivatives required for defining Dirichlet energy and $u|_{\partial\Omega} = g$ are the boundary conditions where $\partial\Omega$ denotes the boundary. The solution to the variational problem in Equation (2.6) is given by the partial diferential equation called Laplace's equation

$$\Delta u = 0 \text{ in } \Omega, \tag{2.8}$$

subjected to the boundary conditions:

$$u = g \quad \text{on } \partial\Omega \tag{2.9}$$

where the differential operator Δ is called the Laplacian. The solution to Laplace's equation provides the smoothest possible interpolation of Dirichlet boundary conditions across the interior of the domain Ω.

Alternatively, one can control the smoothing using the gradient flow of Dirichlet energy which models an evolving field $u(\boldsymbol{x}, \tau)$, starting from the initial condition $u(\boldsymbol{x}, 0)$ and gets smoother over time τ according to the diffusion equation:

$$\frac{\partial u}{\partial \tau} = \dot{u} = \Delta u \quad \text{in} \quad \Omega. \tag{2.10}$$

Notably, the diffusion time τ is a parameter that controls the degree of smoothing, with larger values leading to smoother fields. In our setting, it is not a physical time or wall clock time but rather a mathematical parameter that governs the evolution of the field towards smoothness.

As discussed in the Section 2.2, unlike the flat geometry of Euclidean space, curved manifolds do not admit a global coordinate frame to measure distances, compute angles between tangent vectors, or define differential operators such as the Laplacian. Laplace-Beltrami operator $\Delta_{\mathcal{M}}$ generalizes the Laplacian from Euclidean space to curved spaces but for conciseness, we will refer to Laplace-Beltrami operator as the Laplacian. Accordingly the diffusion equation on a curved manifold $\mathcal{M}$ is given by

$$\dot{u} = \Delta_{\mathcal{M}} u. \tag{2.11}$$

Similarly, for a tangent vector field $\boldsymbol{v}(\boldsymbol{x}, \tau) \in \mathcal{T}_{\boldsymbol{x}}\mathcal{M}$, the governing partial differential equation is defined as:

$$\dot{\boldsymbol{v}} = \Delta_{\nabla} \boldsymbol{v}, \tag{2.12}$$

where Δ_{∇} is the Connection Laplacian. Unlike the scalar Laplace–Beltrami operator, which only requires directional changes of a function, the Connection Laplacian simultaneously track how a vector field changes its spatial distribution and how the underlying coordinate frames rotate due to the manifold's curvature. Notably, in Euclidean space, the Connection Laplacian reduces to the standard Laplacian applied to each vector component.

In practice the Laplacian and the connection Laplacian are only available in closed form for a limited set of highly structured manifolds such as cylinders or spheres. Accordingly, for an arbitrary surface, it is necessary to approximate the Laplacian using a discrete representation of the surface.

2.3.1 Discretized Laplacian on Curved Surfaces

For a discrete surface representation such as a mesh, a point cloud, or a grid with $N_{\mathcal{P}}$ vertices, the discretized Laplacian $\boldsymbol{L} \in \mathbb{R}^{N_{\mathcal{P}} \times N_{\mathcal{P}}}$ can be approximated using the sparse matrices

$$\boldsymbol{L} = \boldsymbol{M}^{-1}\boldsymbol{C}, \tag{2.13}$$

where $\boldsymbol{M}$ is the diagonal mass matrix and $\boldsymbol{C}$ is a sparse symmetric matrix called the weak Laplacian. The entries of $\boldsymbol{M}$ correspond to the Voronoi cell areas in the local tangent plane around each point of $\mathcal{P}$. Similarly, the entries of $\boldsymbol{C}$ are determined by the connectivity of the points on the local tangent space and the distance between the connected points. Note that the local tangent space structure also identifies the boundary points. For a given point, the lines between the original point and its neighbors are constructed. If the angle between two consecutive lines is greater than $\pi/2$, the point is classified as a boundary point, and its boundary condition is set to homogeneous Neumann.

Note that there are various approaches for discretizing the Laplacian on point clouds [84, 11, 63, 20]. In this thesis, we use the approach presented in [84] and the implementation provided by the authors [2]. As explicitly stated in the original paper [84] manifold assumption can be relaxed and the method can be applied to non-manifold point clouds as well.

[2] https://github.com/nmwsharp/robust-laplacians-py.git

2.3.2 Time Discretization of the Diffusion Equation

Next, we discretize the diffusion equation (Eq. (2.11)) in time and incorporate the discrete Laplacian $\boldsymbol{L}$. Using the backward Euler method, we derive the implicit time-stepping equation, which remains stable for any timestep τ

$$\frac{1}{\tau}(\boldsymbol{u}_\tau - \boldsymbol{u}_0) = \boldsymbol{L}\boldsymbol{u}_\tau, \tag{2.14}$$

where $\boldsymbol{u}_0$ and $\boldsymbol{u}_\tau$ are column vectors containing the potential field values at the vertices of the point cloud at the initial and final times, respectively. Then, combining equations (2.13) and (2.14) and solving for $\boldsymbol{u}_\tau$ we obtain the linear system

$$\boldsymbol{u}_\tau = (\boldsymbol{M} - \tau\, \boldsymbol{C})^{-1}\boldsymbol{M}\boldsymbol{u}_0. \tag{2.15}$$

2.3.3 Heat Kernel

Mathematically, the heat kernel is defined as the unique fundamental solution to the diffusion equation under a Dirac delta initial distribution located at $\boldsymbol{y} \in \mathcal{M}$ i.e., $u(\boldsymbol{x}, 0) = \delta_{\boldsymbol{y}}(\boldsymbol{x})$. The heat kernel $K_\tau(\boldsymbol{x}, \boldsymbol{y})$ is defined precisely as the solution to this initial value problem for any destination point $\boldsymbol{x} \in \mathcal{M}$ at diffusion time $\tau > 0$:

$$\begin{cases} \dfrac{\partial K_\tau(\boldsymbol{x}, \boldsymbol{y})}{\partial \tau} = \Delta_{\mathcal{M}} K_\tau(\boldsymbol{x}, \boldsymbol{y}) & \text{for } \tau > 0 \\ \lim_{\tau \to 0^+} K_\tau(\boldsymbol{x}, \boldsymbol{y}) = \delta_{\boldsymbol{y}}(\boldsymbol{x}) & \text{Initial Condition} \end{cases} \tag{2.16}$$

Stochastically, the heat kernel acts as a transition probability density; it describes the probability of a continuous random walker moving from an initial position $\boldsymbol{y}$ to a destination $\boldsymbol{x}$ over an unrolled diffusion time τ.

Utilizing the discrete spectral decomposition of the Laplace–Beltrami operator, this transition density is analytically expressed as an infinite sum of its eigenfunctions:

$$K_\tau(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=0}^{\infty} e^{\lambda_i \tau} \phi_i(\boldsymbol{x}) \phi_i(\boldsymbol{y}) \tag{2.17}$$

where λ_i and ϕ_i represent the eigenvalues and corresponding orthogonal eigenfunctions of $\Delta_{\mathcal{M}}$. In the limiting case where the manifold is a flat, infinite d-dimensional Euclidean space ($\mathcal{M} = \mathbb{R}^d$), the spatial boundaries no longer restrict the random walker. Here, the heat kernel reduces exactly to the classic isotropic Gaussian squared exponential kernel:

$$K_\tau(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{(4\pi\tau)^{d/2}} \exp\left(-\frac{|\boldsymbol{x} - \boldsymbol{y}|^2}{4\tau}\right) \tag{2.18}$$

where the diffusion time parameter maps directly to the spatial variance ($\sigma^2 = 2\tau$). From this perspective, unlike by directly replacing the Euclidean distance with the geodesic distance, the heat kernel can be interpreted as the correct geometry-aware generalization of the Gaussian kernel to non-Euclidean domains [16]. When dealing with point-based representations that violate manifold assumptions, traditional Riemannian geometry fails. From a diffusion geometry perspective [49], the heat kernel resolves this by serving as the primary building block for establishing underlying geometric structure.

Heat Kernel as a Reproducing Kernel

The heat kernel functions as a mathematically rigorous symmetric, positive-definite covariance function that can be utilized to construct a Reproducing Kernel Hilbert Space (RKHS) directly on curved manifolds. This algebraic framing lifts non-Euclidean manipulation data into a high-dimensional Hilbert space, providing a coordinate-free inner product structure. By operating within an RKHS defined by this geometry-aware transition kernel, trajectories and target distributions on the manifold can be smoothed.

Varadhan's Formula

Since Riemannian manifolds are locally Euclidean, Varadhan's asymptotic formula [99] exploits the localized Gaussian reduction in Eq. (2.18) over an infinitesimal time step ($\tau \to 0$). It dictates that before a random walker has time to interact with the wider global curvature or boundaries of the manifold, its short-time probability decay profile can be used to recover the true geodesic distance $d_{\mathcal{M}}(\boldsymbol{x}, \boldsymbol{y})$ across any arbitrary surface:

$$d_{\mathcal{M}}(\boldsymbol{x}, \boldsymbol{y}) = \lim_{\tau \to 0} \sqrt{-4\tau \log K_{\tau}(\boldsymbol{x}, \boldsymbol{y})} \tag{2.19}$$

2.3.4 Heat Method for Geodesics

The Heat Method developed by Crane et al. [26] computes the geodesic distances $d_{\mathcal{M}}(\boldsymbol{x}, \boldsymbol{y})$ from a source point $\boldsymbol{y}$ (or a set of source points) to all other vertices $\boldsymbol{x} \in \mathcal{M}$ on an arbitrary geometric domain. Heat method is a direct application of Varadhan's formula, which states that the geodesic distance can be recovered from the short-time asymptotic behavior of the heat kernel. The key insight is that the direction of the gradient of the heat field, which decays exponentially in magnitude, aligns with the tangent vectors of the true shortest paths across the manifold.

By computing the diffused field using Eq. (2.15) for a tiny fraction of time and normalizing the vector field to discard the decaying magnitude information, one can reconstruct the exact gradient of the distance map $\boldsymbol{X}$. Then, Poisson equation recovers the geodesic distance field ϕ by integrating the gradient field $\boldsymbol{X}$ across the manifold:

$$\Delta_{\mathcal{M}} \phi = \mathrm{div}_{\mathcal{M}}(\boldsymbol{X})$$

which reduces to solving another sparse linear system in the discrete setting.

2.3.5 Vector Heat Method for Parallel Transport

The Vector Heat Method extends Varadhan's localized scalar reasoning to connection calculus, providing a coordinate-free approach to evaluate the parallel transport of a tangent vector $\boldsymbol{v}_0 \in \mathcal{T}_{\boldsymbol{y}}\mathcal{M}$ from a source coordinate $\boldsymbol{y}$ globally across the entire manifold.

Sharp et al. [85], used Varadhan's Formula to show that short-time vector diffusion under the connection matches the true parallel-transport vector field, similar to the geodesic Heat method. The algorithm consists of vector diffusion (2.12) followed by a magnitude correction step relying on scalar diffusion (2.11) to restore the true scale of the parallel-transported vector field.

2.4 Ergodic Control

Here we introduce the ergodic control from the perspective of diffusion processes. Ergodicity describes the requirement that a dynamical system's trajectory spends time in regions of a space in proportion to a desired density. This concept is particularly relevant to exploration tasks in which the goal is to ensure that a robot or agent effectively samples an environment according to a specified distribution. At its core, *diffusion processes* are the mechanism that drives a system toward ergodicity.

The Empirical Distribution

Consider a dynamical system with a state trajectory $s(t)$. Over a time interval $[0, t]$, the system generates an empirical distribution, hereafter referred to as the coverage, within the task space $\mathcal{X}$. This is defined via a mapping h from the state space to the task space as follows:

$$p_s(t,x) = \frac{1}{t}\int_0^t \delta(x - h(s(\tau)))\,d\tau \tag{2.20}$$

where $\delta(\cdot)$ denotes the Dirac delta function. Essentially, $p_s(t,x)$ represents the time-averaged spatial occupancy of the system up to time t.

Ergodicity

A system is defined as ergodic with respect to a target distribution $q(x) \in \mathcal{P}(\mathcal{X})$ if its coverage converges to the target distribution in the limit

$$\lim_{T\to\infty} \mathbb{E}_{x\sim p_{s,T}}[\phi(x)] = \mathbb{E}_{x\sim q}[\phi(x)], \quad \forall \phi \in \mathcal{C}(\mathcal{X}). \tag{2.21}$$

Formally, this convergence is in distribution meaning that for any continuous test function $\phi(x)$, the time-average of the function along the trajectory equals the spatial average of the function over the target distribution.

Ergodic Control Objective

The objective of ergodic control is to derive a control law $u^*(t)$ such that the resulting coverage p_s minimizes a discrepancy measure $\mathscr{D} : \mathscr{P}(\mathscr{X}) \times \mathscr{P}(\mathscr{X}) \to \mathbb{R}_{\geq 0}$ relative to the target distribution $q(x)$

$$u^*(t) = \underset{u(t)}{\arg\min} \mathscr{D}\left(p_s(x, T), q(x)\right) \tag{2.22}$$

$$\text{s.t. } s(t) = s_0 + \int_0^t f(s(t'), u(t'))dt'. \tag{2.23}$$

Common choices for this discrepancy measure include Fourier ergodic metric [69], Kullback-Leibler divergence [4] and kernel ergodic metric [93]. In this thesis, we primarily focus on one-step, model predictive control laws, which are derived by minimizing the instantaneous change in the ergodic metric at each time step. We prefer this control design over trajectory-optimization-based approaches because it is highly reactive and robust to model inaccuracies and disturbances, while remaining inherently free of getting trapped in spurious attractors thanks to the properties of ergodicity.

2.4.1 Diffusion Processes for Ergodic Control

Diffusion processes provide a natural mechanism to drive a system toward ergodicity. Here we provide the density control perspective on how diffusion processes can be used to drive a system towards ergodicity [78] by using the Fokker-Planck equation (macroscopic PDE version of stochastic Langevin dynamics), which describes the time evolution of the probability density function $p(x, t)$ of a stochastic process under the influence of a drift term $\boldsymbol{v}$ and a diffusion term

$$\frac{\partial p}{\partial \tau} = -\nabla \cdot (\boldsymbol{v} p) + \Delta p. \tag{2.24}$$

Notably, the Fokker-Planck equation combines the continuity equation used in flow matching [59] and the diffusion equation (2.10). We assume as time goes to infinity $\tau \to \infty$ there exists a stationary distribution p_∞ such that $\frac{\partial p_\infty}{\partial \tau} = 0$, then the Fokker-Planck equation reduces to

$$\Delta p_\infty = \nabla \cdot (\boldsymbol{v} p_\infty). \tag{2.25}$$

We know that the Laplacian Δ is the divergence of the gradient, so we can rewrite the above equation as

$$\nabla \cdot (\nabla p_\infty - \boldsymbol{v} p_\infty) = 0. \tag{2.26}$$

This condition dictates that the net probability flux is entirely divergence-free, preventing the system from trapping trajectories in local sinks and ensuring ergodic coverage of the state space. By designing the drift velocity as the gradient of the log-target density: $\boldsymbol{v} = \nabla \log q$ the stationary distribution p_∞ is mathematically guaranteed to converge to the target distribution $q(x)$ [78]. In modern generative modeling [38], this steering vector field $\nabla \log q(x)$ is defined as the score function. Thus, Langevin dynamics achieves ergodicity stochastically by balancing a score-driven drift with diffusion.

2.4.2 Heat Equation-Driven Area Coverage (HEDAC)

Although score-based Langevin dynamics provides a foundation for ergodic sampling, it lacks a direct mechanism to account for deterministic system dynamics or keeping track of coverage history. Heat Equation-Driven Area Coverage (HEDAC) bridges this gap by translating the stochastic, score-driven framework into a fully deterministic closed-loop control law.

HEDAC uses a source term[3] $\mu(x,t)$ encoding the target spatial distribution and already covered regions

$$\mu(x,t) = q(x)\exp(-p_s(x,t)). \tag{2.27}$$

Using the score of this source term $\nabla \log \mu(x,t)$ as the control command similarly to Langevin dynamics would indeed drive the system towards the target distribution but due to the non-smooth nature of the empirical coverage $p_s(x,t)$, this would lead to a discontinous vector field with spurious attractors. Instead, HEDAC smooths the source term using a stationary diffusion process

$$k\Delta u(x) - u(x) + \mu(x,t) = 0, \tag{2.28}$$

called the screened Poisson equation, where k is a tunable parameter controlling the degree of smoothing. Notably, the $-u(x)$ term helps to balance the influence of the source term. The boundary behavior is governed by a zero-Neumann condition:

$$\mathbf{n}\nabla u(x) = 0 \quad \forall x \in \partial\Omega, \tag{2.29}$$

where $\mathbf{n}$ is the outward unit normal vector to the boundary $\partial\Omega$. This boundary condition ensures that the potential field does not induce motion across the domain boundaries. The control command is then given by the gradient of this smoothed potential field $u(x)$,

$$\dot{s}(t) = \nabla u(s(t),t), \tag{2.30}$$

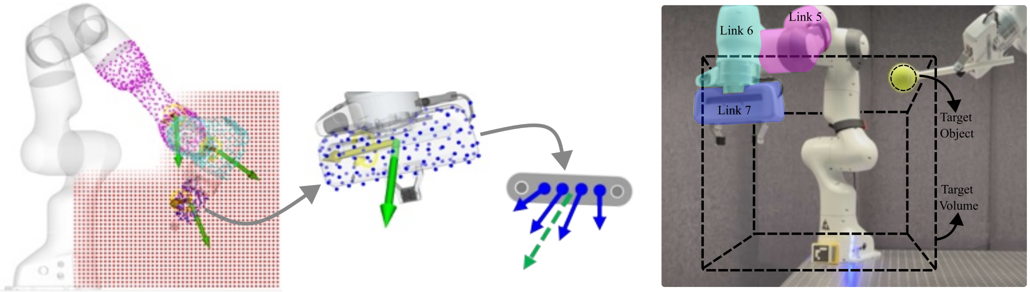
which effectively guides the system toward regions of high target density while avoiding already covered areas. By iteratively updating the coverage distribution and solving the PDE, HEDAC

[3] Original HEDAC [44] used a different source term but it is updated to this formulation in later papers [42].

continuously adapts the control policy to achieve ergodic coverage over time.

3 Whole-Body Ergodic Exploration using Diffusion

Whole-body exploration Individual exploration agents Real-world experiments



This chapter introduces a whole-body ergodic control method that utilizes a multi-link robot manipulator as a continuous tactile sensor for spatial exploration. Ergodic control is traditionally restricted to point-mass agents. We extend this framework to the complex geometry of a multi-link robot manipulator by leveraging controlled smoothness through diffusion processes as an inductive bias for whole-body coordination.

Publication Note

The material presented in this chapter is adapted from the following publication.

- **Bilaloglu, C.**, Löw, T., and Calinon, S. (2023). Whole-Body Ergodic Exploration With a Manipulator Using Diffusion. *IEEE Robotics and Automation Letters*

Supplementary Material

Supplementary materials related to this chapter are available at: https://sites.google.com/view/w-ee-d

3.1 Introduction

Tactile exploration is essential when far-field sensors (e.g., cameras and LiDAR) are occluded, lack millimeter-level precision, or fail to capture surface properties such as stiffness and friction. In these scenarios, joint torque sensors and tactile skins [72] can turn the entire surface of a robot's kinematic chain into a distributed tactile sensor. This approach can be leveraged across various platforms, including manipulators [2], multi-fingered hands [88], and legged robots. However, current ergodic control methods cannot exploit this extended sensor footprint. Instead, they remain restricted to simplified footprints, such as point sensors [68], radial-basis functions [44], or beam models [4].

In this chapter, we leverage the robot's full kinematic chain and link surfaces to maximize sensor footprint during tactile exploration. We achieve this by decomposing the *whole-body* of the robot (combining all links) into a set of kinematically constrained *virtual agents*. The primary challenge lies in formulating a locally consistent exploration behavior that coordinates the motion of individual agents considering the global exploration objective of the robot. Inspired by HEDAC, we formulate this coordination using a smooth task-space field that guides the virtual exploration agents. However instead of solving a stationary diffusion PDE we use a non-stationary diffusion process to have more control over local smoothness of the field. Accordingly, nearby virtual agents are encouraged to explore similar regions of the space, where as more distant agents on different links can explore different regions. We further enforce this by introducing weighting strategies at both the agent and link levels to prioritize the exploration oflarger and distal links closer to the target. Consequently, we present the first whole-body ergodic exploration method and the first three-dimensional implementation of the HEDAC. Our contributions are as follows:

- modeling the whole-body as a collection of virtual exploration agents constrained by the robot's kinematics;
- formulating a locally smooth field to coordinate link-level virtual agents for coherent whole-body exploration;
- frontier, manipulability, and link volume-based weighting to prioritize larger and distal links closer to the target.

3.2 Method

We represent the whole-body of the robot as a collection of virtual agents that are kinematically constrained to move with the robot's links. In order to combine individual exploration behaviors of these agents into a coherent whole-body exploration strategy, we design a control law that ensures nearby agents have similar exploration goals. We achieve that by enforcing spatial smoothness in the task-space field that guides the exploration behavior of these agents.

We enforce smoothness by using a diffusion process to generate the potential field that guides the exploration behavior of these agents.

3.2.1 Locally Smooth Diffused Fields

We modify HEDAC to obtain the potential field guiding the exploration behavior. Original implementation [44] solves stationary ($\dot{u}(\boldsymbol{x},\tau)=0$) diffusion PDE given in Eq. (2.28). Since HEDAC solves the stationary diffusion equation, the resulting potential field is globally smooth across the domain. However, this global smoothness can result in losing the local details which can hinder link-level coordination. Therefore, we slow down the diffusion process by solving non-stationary ($\dot{u}(\boldsymbol{x},\tau)\neq 0$) diffusion to increase local exploration

$$\dot{u}(\boldsymbol{x},\tau)=\alpha\cdot\Delta u(\boldsymbol{x},\tau)-u(\mathbf{x},\tau)+\mu(\boldsymbol{x},t). \tag{3.1}$$

Note that, τ is the diffusion time parameter, and t is the actual time of the system. The source term $\mu(\boldsymbol{x},t)$ is a function of the current coverage $p_s(\boldsymbol{x},t)$ and the target distribution $q(\boldsymbol{x})$ given in Eq. (2.27). We numerically integrate Eq. (3.1) using an explicit time-stepping scheme

$$u(\boldsymbol{x},\tau+1)=u(\boldsymbol{x},\tau)+\dot{u}(\boldsymbol{x},\tau)\cdot\delta\tau, \tag{3.2}$$

where the maximum stable timestep $\delta\tau$ is a function of the diffusion rate α and given by Courant–Friedrichs–Lewy condition

$$\delta\tau\left(\sum_{i=1}^{N_d}\frac{\alpha_i}{\delta x_i}\right)\leq 1, \tag{3.3}$$

with the spatial discretization δx and the number of spatial dimensions N_d. We introduce the parameter N_k, for the number of integration steps we take using the maximum stepsize $\delta\tau$ computed by Eq. (3.3). Note that choosing high values for N_k would mean integrating Eq. (3.1) until $u(\boldsymbol{x},\tau+1)-u(\boldsymbol{x},\tau)\approx 0$ and would be equivalent to stationary diffusion.

3.2.2 Kinematically Constrained Virtual Agents

We define *virtual agents* as atomic particles that compose a rigid body and interact with the potential field $u(\boldsymbol{x},t)$. Depending on the task, virtual agents abstract a sensor/tool used for physical interaction during exploration. Additionally, we use the term *whole-body* if the sensor/tool spans multiple bodies on different links of the manipulator. By construction, we can locate each agent in the potential field using the forward kinematics function f_{kin} of the robot

$$\boldsymbol{x}_{i,j}=\boldsymbol{f}_{\text{kin}}(\boldsymbol{q},i,j)\quad\forall i=1,\ldots,N_j\quad\forall j=1,\ldots,M, \tag{3.4}$$

where N_j is the number of virtual agents on the j-th link, and M is the number of links composing the whole-body. $\boldsymbol{x}_{i,j}$ is the position of the i-th agent on the j-th link and $\boldsymbol{q}$ is the vector of joint variables.

3.2.3 Active Agents and Local Weighting

We call the virtual agents that we use for computing coverage but not contributing to the control action as *passive*. We use the term *passive* because these agents explore regions indirectly. Still considering their coverage relieves the need to revisit those locations later. On the other hand, *active* agents, contribute to the control command of the robot with their local exploration goal in addition to coverage.

According to our model, the potential field exerts a fictitious force on each active agent based on the gradient of the potential field, and we multiply this force by an agent weight $w_{i,j}$

$$\boldsymbol{f}_{i,j} = w_{i,j} \nabla u(\boldsymbol{x}_{i,j}(t), \tau). \tag{3.5}$$

The naive method of computing the agent weights is to use a uniform weighting strategy, thus assigning equal importance to every agent. However, this is suboptimal since the significance of agents differs depending on their position in the potential field and the current state of the potential field itself. The value of the potential field at a given point encodes how much this particular region is underexplored. Accordingly, we embed this information by using the local value sensed by the agent as its weight. Hence, agents on the frontier of exploration (the ones closer to the underexplored regions) will have a higher weight than those on the overexplored regions. Thus, we set the normalized weight of the i-th agent on the j-th link as

$$w_{i,j} = \frac{\tilde{w}_{i,j}}{\sum_{i=1}^{N_j} \tilde{w}_{i,j}} \quad \text{with} \quad \tilde{w}_{i,j} = u(\boldsymbol{x}_{i,j}(t), \tau), \tag{3.6}$$

and call it *local weighting strategy*. Note that, the local weight is a function of the potential field, i.e. both space and time, and is therefore computed online. We show the difference between local and uniform weighting strategies in Figure 3.1.

3.2.4 Active Links and Manipulability Weighting

Similarly to active agents, we call a rigid body composed of active agents *active link* and compute the net force and moment acting on it by all the agents

$$\boldsymbol{f}_{\text{net}_j} = \sum_{i=1}^{N_j} \boldsymbol{f}_{i,j}, \quad \boldsymbol{m}_{\text{net}_j} = \sum_{i=1}^{N_j} \boldsymbol{r}_{i,j} \times \boldsymbol{f}_{i,j}, \tag{3.7}$$

where $\boldsymbol{r}_{i,j}$ is the displacement vector connecting the active agent and the j-th link's center of mass. We concatenate force and moment into a net wrench acting on the j-th link of the manipulator. For the simplest kinematic control strategy, we set the desired twist of the link

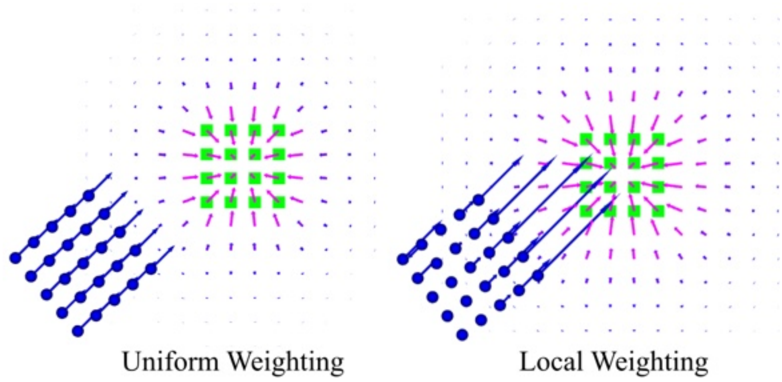


Figure 3.1: Comparison of uniform and local weighting. The green square is the exploration target and small arrows show the gradient of the diffused field. Blue dots and arrows show active agents and the force exerted on each agent after weighting.

$\boldsymbol{v}_{\text{des}_j}$ equal to the net wrench acting on the j-th link

$$\boldsymbol{v}_{\text{des}_j} = \begin{bmatrix} \boldsymbol{f}_{\text{net}_j}^{\top} & \boldsymbol{m}_{\text{net}_j}^{\top} \end{bmatrix}^{\top} \tag{3.8}$$

corresponding to having identity inertia.

Here the twist commands that we generate correspond to consensus among the agents because the gradient field exerts similar forces to neighboring agents (local consistency) since we set the local cooling term used for collision avoidance in HEDAC to zero and slowed down diffusion by solving the non-stationary diffusion equation. Moreover, the gradient (force) field resulting from spatial diffusion is smooth, and we perform first local exploration (moving based on the force field) and then global exploration (by the propagation of information as diffusion).

We weight the contribution of the active links to the control command because each link has different *manipulability* [108] and volume v, resulting in distinct volumetric coverage rates. For that purpose, we compute the link weights using the scalar manipulability index μ

$$w_j = v_j \mu_j, \tag{3.9}$$

$$\text{with} \quad \mu_j = \sqrt{\det\left(\boldsymbol{J}_j(\boldsymbol{q})\boldsymbol{J}_j(\boldsymbol{q})^{\top}\right)}, \tag{3.10}$$

where $\boldsymbol{J}_j(\boldsymbol{q})$ is the Jacobian of the j-th active link computed at its center of mass.

3.2.5 Consensus Control for Whole-Body

Our setting consists of multiple weight-prioritized tasks, encoded as task velocities with corresponding Jacobians, and we would like to perform them in the least square optimal sense by exploiting the redundancy of our robot

$$\dot{\boldsymbol{q}}_{\text{des}} = \arg\min_{\dot{\boldsymbol{q}}} \, (\bar{\boldsymbol{v}}_{\text{des}} - \bar{\boldsymbol{J}}\dot{\boldsymbol{q}})^{\top} \boldsymbol{W} (\bar{\boldsymbol{v}}_{\text{des}} - \bar{\boldsymbol{J}}\dot{\boldsymbol{q}}) \tag{3.11}$$

$$\text{with} \quad \boldsymbol{W} = \text{blockdiag}\big(\boldsymbol{I}w_1, \boldsymbol{I}w_2, \ldots, \boldsymbol{I}w_m\big) \tag{3.12}$$

where $\bar{\boldsymbol{J}}$ is the vertical concatenation of the Jacobians, and $\bar{\boldsymbol{v}}_{\text{des}}$ is the vertical concatenation of the task velocities of active links. We enforce task priorities by using the weighted pseudoinverse and computing the desired joint velocities as

$$\dot{\boldsymbol{q}}_{\text{des}} = \big(\bar{\boldsymbol{J}}^{\top} \boldsymbol{W} \bar{\boldsymbol{J}}\big)^{-1} \bar{\boldsymbol{J}}^{\top} \boldsymbol{W} \bar{\boldsymbol{v}}_{\text{des}}. \tag{3.13}$$

Next, we use desired joint velocity to kinematically simulate the robot $\boldsymbol{q}_{t+1} = \boldsymbol{q}_t + \dot{\boldsymbol{q}}_{\text{des}} \cdot \Delta t$. Then, we clamp the desired joint positions as the simplest strategy to comply with joint limits and give them as targets to a joint impedance controller. We give the full procedure for robot control in Algorithm 1.

Algorithm 1: Whole-Body Exploration

Input: target distribution $p(\boldsymbol{x})$, forward kinematics function $\boldsymbol{f}_{\text{kin}}(\boldsymbol{q}, i, j)$, initial joint configuration $\boldsymbol{q}_0$, number of active links $\boldsymbol{M}$, number of active agents $\boldsymbol{N_j}$ in j-th link

Output: control commands $\boldsymbol{u}(t)$

initialize agents and bodies using using Eq. (3.4) and $\boldsymbol{q}_0$

compute δt using Eq. (3.3)

while *true* **do**

  compute $c(\boldsymbol{x}, t)$ and $s(\boldsymbol{x}, t)$ as in [44]

  for $k \leftarrow 1$ *to* N_k **do**

    integrate Eq. (3.1) using Eq. (3.2)

  for *all* M *bodies* **do**

    for *all* N_j *agents* **do**

      compute $\boldsymbol{x}_{i,j}$ and $\boldsymbol{f}_{i,j}$ using Eq. (3.4) and Eq. (3.5)

    compute $\boldsymbol{v}_{\text{des}_j}$ and w_j using Eq. (3.8) and Eq. (3.10)

  compute $\dot{\boldsymbol{q}}_{\text{des}}$ using Eq. (3.13)

3.3 Experiments

3.3.1 Simulated Experiments

We performed kinematic simulations to measure the exploration performance. We used the normalized ergodicity over the target distribution as the exploration metric

$$\varepsilon = \frac{\| \max\big((\boldsymbol{p} - \boldsymbol{c}_t), 0\big) \|_2}{\sum_{\Omega} p_{i,j}}, \tag{3.14}$$

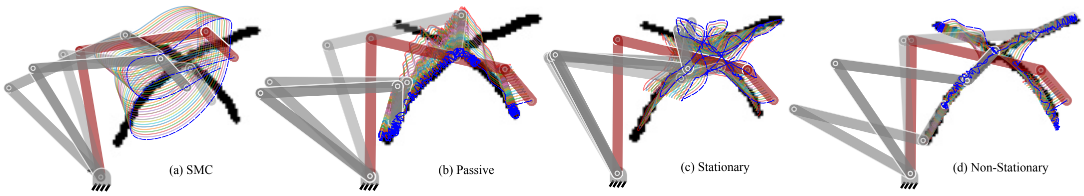


Figure 3.2: Planar exploration using different configurations. The black shape is the target distribution, and the colored lines are agent trajectories, where the blue dashed lines correspond to the agent's path at the tip of the last link. We show the configuration of the planar manipulator at equally spaced timesteps. Red is the start configuration, and the transparency decreases as time increases.

where $\boldsymbol{p}$ is the discrete target distribution, $\boldsymbol{c}_t$ is the discrete coverage distribution calculated as in [44]. This metric shows how good the time-averaged statistics of agent trajectories match the target distribution, and lower values indicate higher performance.

Planar Experiments

We first present the results of planar simulations tested in four different settings: (i) *SMC*, (ii) *passive*, (iii) *active/stationary* using Eq. (2.28), and (iv) *active/non-stationary* ($N_k = 1$) using Eq. (3.1). SMC and passive configurations use a single active agent at the tip of the last link that we control by the state-of-the-art ergodic control methods SMC and HEDAC, respectively, where the passive configuration also considers the coverage of the last link indirectly for the control. We used active agents sampled on the last link equally spaced by the unit distance for the active configurations. We computed the metric given in Eq. (3.14) using the coverage of the last link for all of these configurations. We used a 75×75 grid for discretizing the exploration domain and 20 basis functions for the SMC method.

We plotted the trajectories of a planar 3-link manipulator under these different configurations in Fig. 3.2. We observe that although SMC and passive configurations fail to track the target effectively, the stationary diffusion approach exhibits limited alignment. Conversely, under non-stationary diffusion, agents remain aligned with the target distribution for the majority of their trajectories, with brief misalignments occurring only during transient re-orientation phases. This behavior is expected, as non-stationary diffusion intensifies local exploration and enhances agent coordination when compared to stationary diffusion.

To quantify these observations, we repeated the same experiment for two different target distributions starting from 100 uniformly sampled initial joint configurations. We chose one diffuse and one fine-detailed target distribution to test the scenarios advantaging stationary/non-stationary diffusion, respectively. Although the exploration behavior would continue indefinitely, we stopped the simulation after $t = 1000$ timesteps. We calculated the mean and standard deviation of the normalized ergodic metric given in Eq. (3.14) and plotted the results in Fig. 3.3.

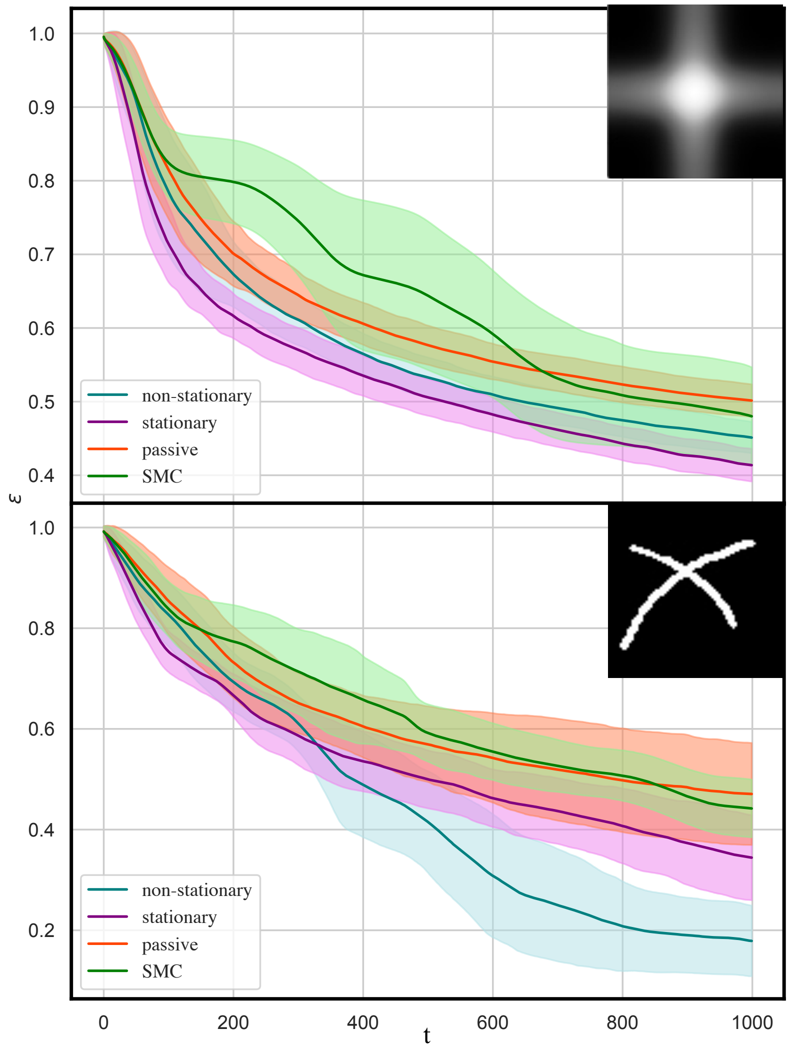


Figure 3.3: Coverage performance given by the normalized ergodic metric for different virtual agent configurations. Target distributions for the coverage task are given on the top right.

These results corroborate our qualitative findings, demonstrating that active agents outperform both the state-of-the-art SMC method and passive configurations. However, the performance gap between stationary and non-stationary diffusion is sensitive to the exploration duration and the specific geometry of the target distribution. Specifically, stationary diffusion ($N_k \gg 1$) over-smooths the target field; although this facilitates coarse exploration of diffuse targets, it fails to resolve distal or high-frequency spatial details. In contrast, non-stationary diffusion ($N_k = 1$) significantly improves performance in finegrained and distal regions, albeit with a slight performance trade-off for diffuse targets. Results for intermediate integration steps ($N_k \in (1, 10]$) effectively interpolate the performance characteristics of the stationary and non-stationary extremes shown in Fig. 3.3. Consequently, we recommend a single-step approach ($N_k = 1$) for concentrated targets, higher values ($N_k \approx 10$) for diffuse distributions, and low-to-medium values ($N_k \approx 3$) for general-purpose exploration.

Three Dimensional Experiments

In the first 3D experiment, we measured the effect of using multiple active links on the exploration performance. We assumed a uniform exploration target and used a cube discretized on a $50 \times 50 \times 50$ grid where each point corresponds to 1 cm as the target distribution. We placed the target in front of a 7-axis Franka Emika robot and sampled active agents on links 5, 6, and 7 using Poisson-disk sampling. Note that we used the same agent configuration for computing the coverage and the metric using Eq. (3.14), but we changed the number of active links used for computing robot control commands. Similarly to the planar experiments, we used 30 pre-sampled initial joint configurations and plotted the results in Figure 3.4.

The 3D experiments indicate that leveraging multiple links enhances exploration performance. However, these gains diminish as we move toward proximal links with reduced manipulability. Links closer to the base, specifically link 5, receive negligible weights during the control command normalization process. Consequently, these links function as effectively passive agents; while they contribute to total volumetric coverage, they exert minimal influence on the robot's active control commands.

This phenomenon partly explains the marginal performance gains observed beyond link 6. We attribute this plateau primarily to the scale of the target distribution relative to the link dimensions. When the target is small, the advantages of an expanded sensor footprint are inherently limited. While the workspace in this configuration was constrained by the manipulator's fixed base and joint limits, we posit that the correlation between sensor footprint and performance would strengthen in systems with a lower link-to-target size ratio. Promising applications for this whole-body approach include mobile manipulation, where the reachable exploration area is significantly larger, or high-DoF robotic hands, where the individual links are small relative to the target objects.

Secondly, we compared the exploration performance of the proposed method to that of using a search pattern. In this task, the robot explored the same target region using its whole-body

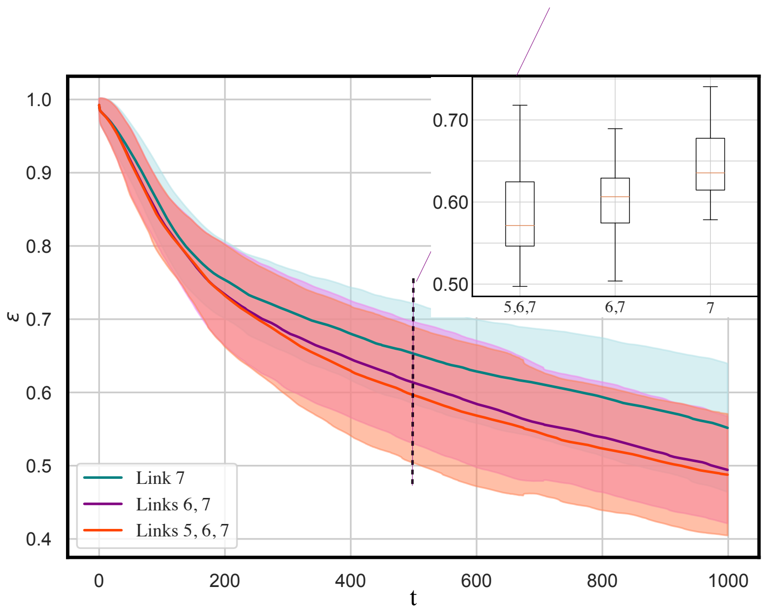
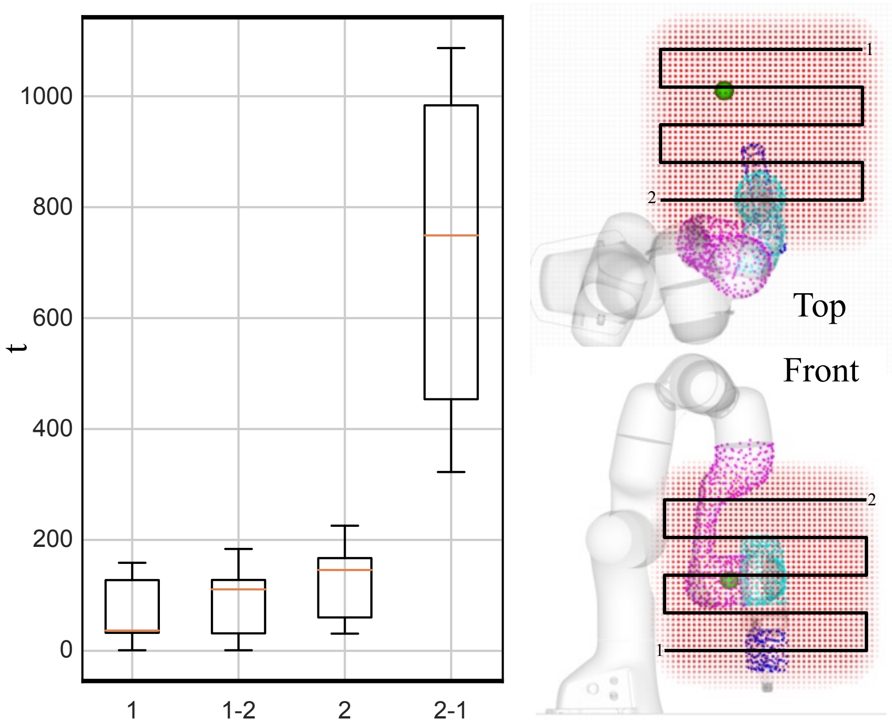


Figure 3.4: Coverage performance given by the normalized ergodic metric for different virtual agent configurations. The inset boxplot shows the performance in detail for each configuration at 500-th timestep. During the experiments, we set $N_k = 3$ as an empirically found moderate value and attained a control frequency of 35, 18, and 13 Hz for active agents on the last one, two, and three links respectively using a laptop processor.

composed of links 5, 6, and 7 until one of the links touched the sphere target placed at an unknown position. We repeated the experiment 30 times for randomly sampled target sphere positions but used fixed initial configurations corresponding to the end-effector poses 1, 2 given in Fig. 3.5b. We recorded the time until the first contact and plotted the results in Fig. 3.5a. In the “Path 1-2” scenario, links 5, 6, and 7 are initially immersed within the target distribution; conversely, in the “Path 2-1” scenario, the links only engage with the target area at the end of the trajectory. These two paths represent the best- and worst-case initialization scenarios for the planned search. Our results demonstrate that the proposed whole-body ergodic approach is significantly more robust to the initial robot configuration, outperforming even the best-case planned path (1-2) by more effectively leveraging the robot’s full kinematic chain.

3.3.2 Real-world Experiment

For the real-world experiments, we used a 7-axis Franka Emika robot with the same object localization task performed in kinematic simulation. We placed a tennis ball inside the target distribution as the target object at an unknown location. We first recorded the trajectory resulting from the exploration behavior and then tracked the trajectory with a stiff impedance controller to ensure that the robot safely contacted the tennis ball and did not hit the second robot or the stick. We ran the experiment until one of the links made contact with the target



(a) (b)

Figure 3.5: Coverage performance given by the timesteps required until the first contact of the whole-body with the target object (green sphere). a) The horizontal axis of the boxplot corresponds to our method and a planned search pattern starting from the initial end-effector poses 1 and 2. b) We show the initial poses 1, 2 and the direction of the search pattern is indicated by the ordering of the poses.

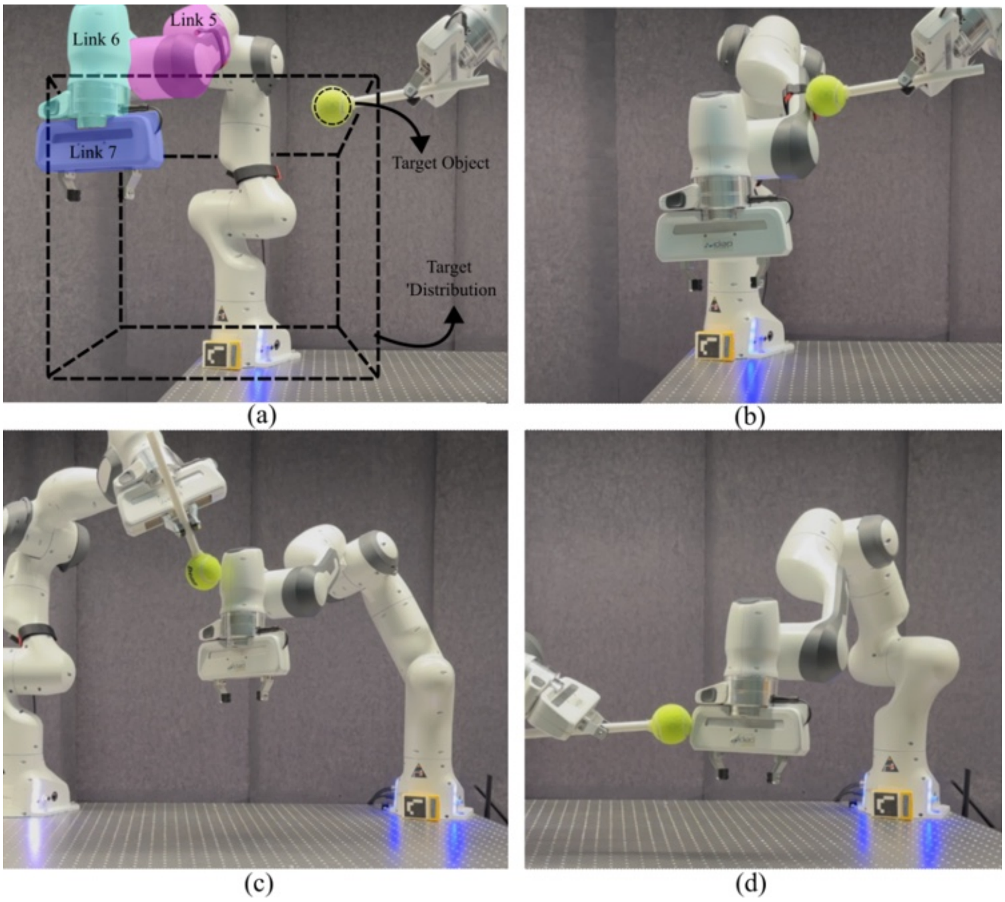


Figure 3.6: Real-world experiment of the robot exploring the cube in dashed lines using its last three links, until either b) link 5, c) link 6 or d) link 7 contacts the target object.

object, which we detected by using the joint torque sensors of the robot. We provide the experiment setup in Figure 3.6.

We showcased the method's applicability with a manipulator in a real-world experiment and presented the first 3D control implementation of the HEDAC approach. Real-world experiments also showed we can perform whole-body tactile exploration using off-the-shelf torque-controlled robots without additional sensors.

3.4 Conclusion

We presented a control method for efficient spatial exploration utilizing a robotic manipulator's entire kinematic chain. By leveraging the controlled smoothness inherent in the diffusion process, we established a spatio-temporal coordination field for the virtual exploration agents that compose the robot's whole-body. Comparative analysis with existing benchmarks demonstrated the superior performance and robustness of this diffusion-based approach. Furthermore, physical experiments validated that the method enables high-fidelity tactile exploration using only intrinsic torque sensing, without the need for specialized external sensors.

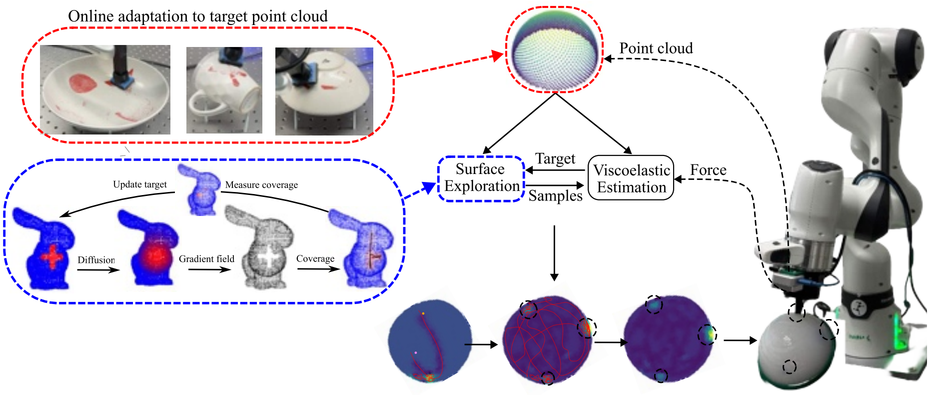
3.4.1 Limitations and Future Work

These results suggest a broad applicability to other proprioceptive platforms, such as legged robots and multi-fingered hands. For instance, the proposed method could enable legged systems to actively explore ground contact points or allow robotic hands to execute contact-rich grasping maneuvers where fingers autonomously resolve object geometry. While joint torque sensing effectively facilitates contact detection, we acknowledge that precise spatial localization on the robot's surface remains a challenge, one that could be addressed in future work through the integration of high-resolution whole-body tactile skins.

The primary limitation of the current whole-body exploration method is the restricted workspace imposed by the fixed base and joint limits. Transitioning to a mobile manipulator platform would significantly enhance performance by permitting a more favorable ratio between link dimensions and the spatial extent of the target distribution. On such a platform, even proximal links with lower manipulability could be leveraged effectively due to their large surface area and increased mobility.

However, extending this approach to mobile manipulation introduces a significant computational challenge: the increased dimensionality and domain size required to solve the diffusion equation. As the workspace scales, traditional grid-based solvers become computationally prohibitive. To address this, future research could investigate Monte Carlo methods such as the Walk-on-Spheres [79] algorithm, which allows for point-wise evaluation of the solution without a global grid. Additionally, data-driven or low-rank approximations (Tensor Train decomposition [87] or Physics-Informed Neural Networks) offer promising avenues for efficiently approximating the diffusion field in high-dimensional or large-scale environments.

4 Ergodic Control and Mapping on Curved Surfaces using Diffusion



This chapter introduces a geometric approach for online exploration, coverage and mapping of physical parameters on curved surfaces. By exploiting intrinsic diffusion processes, we extend ergodic control to curved surfaces measured online as point clouds. We demonstrate the versatility of this approach through real-world robotic tasks: tactile coverage in surface-cleaning experiments with non-uniform distributions, and tactile exploration for viscoelastic mapping of curved, deformable surfaces.

Publication Note

The material presented in this chapter is adapted from the following publication.

- **Bilaloglu, C.***, Löw, T.*, and Calinon, S. (2025). Tactile Ergodic Coverage on Curved Surfaces. *IEEE Transactions on Robotics*
- Beber, L.*, **Bilaloglu, C.***, Lamon, E., Saveriano, M., Fontanelli, D., Palopoli, L., & Calinon, S. (2026). Autonomous Robotic Palpation for Viscoelastic Mapping of Curved Soft-tissues. (*Under Review*)

Supplementary Material

Supplementary materials related to this chapter are available at:
https://sites.google.com/view/tactile-ergodic-control/

4.1 Introduction

A wide range of manipulation tasks across domestic, industrial, and medical domains such as surface cleaning and sanding [54, 3], medical palpation [5, 105], and autonomous tactile dataset collection [55] can be formulated as tactile exploration or coverage problems.

Ergodic control provides a principled framework for both exploration and coverage, but existing approaches are largely restricted to non-tactile settings with far-field sensors and Euclidean domains with simple boundaries or homogeneous manifolds. Jacobs et al. [46] extended the Spectral Multiscale Coverage (SMC) algorithm to homogeneous manifolds like spheres and tori using closed-form expressions for Laplacian eigenfunctions. More recently, Sun et al. introduced the kernel ergodic metric [93] to improve computational scalability and extend capabilities to Lie groups. However, arbitrary curved surfaces captured by vision sensors, typically represented as raw point clouds, lack both group structures and homogeneous manifold properties. While the HEDAC method has been adapted to planar meshes with obstacles [45], maze exploration [27], and coverage path planning on non-planar meshes [43], its application to complex curved surfaces remains limited to watertight meshes and offline execution due to preprocessing overhead.

To address these limitations, we formulate an online tactile ergodic controller that operates directly on point clouds. Utilizing point cloud representations allows the system to acquire object surface and target spatial distribution at runtime via vision, track exploration progress using vision data as a proxy signal to measure coverage progress, and compensate for unmodeled task dynamics. We achieve this by solving the diffusion PDE directly on the point cloud in real-time.

For tactile exploration tasks that involve mapping physical parameters like surface stiffness on the surface, we close the loop with an estimation module governed by Gaussian processes utilizing the point cloud's heat kernel. Notably, both the exploration and estimation modules share a unified structural foundation derived from diffusion processes. In summary, our core contributions are:

- providing a robust formulation for difficult-to-model tactile tasks such as cleaning using ergodic control on point clouds
- extending ergodic control to arbitrary curved surfaces using diffusion processes on point clouds,
- re-using the Laplacian operator for Gaussian Processes on point clouds and closing the loop for mapping physical parameters on curved surfaces.

4.2 Background

4.2.1 Gaussian Processes

We provide a brief introduction to Gaussian processes (GPs) as they are used in this chapter for estimating spatial distributions. A Gaussian process is a Bayesian non-parametric model for an unknown function $f(\boldsymbol{x})$, providing both predictions and uncertainty estimates

$$f(\boldsymbol{x}) \sim \mathrm{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')), \tag{4.1}$$

where $m(\boldsymbol{x})$ is the mean function and $k(\boldsymbol{x}, \boldsymbol{x}')$ is the covariance (kernel) function

$$m(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})], \tag{4.2}$$
$$k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}[(f(\boldsymbol{x}) - m(\boldsymbol{x}))(f(\boldsymbol{x}') - m(\boldsymbol{x}'))]. \tag{4.3}$$

Given training data $\mathscr{D} = \{(\boldsymbol{X}, \boldsymbol{y})\}$ with $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]^\top$ and $\boldsymbol{y} = [y_1, \ldots, y_n]^\top$, the GP model assumes a zero mean Gaussian observation model:

$$\boldsymbol{y} = f(\boldsymbol{X}) + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma_n^2 \boldsymbol{I}) \tag{4.4}$$

Then, the joint distribution of the training outputs $\boldsymbol{y}$ and the test outputs $f_* = f(\boldsymbol{X}_*)$ is

$$\begin{bmatrix} \boldsymbol{y} \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{m}_X \\ \boldsymbol{m}_* \end{bmatrix}, \begin{bmatrix} \boldsymbol{A} & K_{X*} \\ K_{*X} & K_{**} \end{bmatrix} \right). \tag{4.5}$$

where

$$\boldsymbol{m}_X := m(\boldsymbol{X}), \qquad \boldsymbol{m}_* := m(\boldsymbol{X}_*), \tag{4.6}$$
$$K_{XX} := K(\boldsymbol{X}, \boldsymbol{X}), \qquad K_{X*} := K(\boldsymbol{X}, \boldsymbol{X}_*), \tag{4.7}$$
$$K_{*X} := K(\boldsymbol{X}_*, \boldsymbol{X}), \qquad K_{**} := K(\boldsymbol{X}_*, \boldsymbol{X}_*), \tag{4.8}$$
$$\boldsymbol{A} := K_{XX} + \sigma_n^2 \boldsymbol{I}. \tag{4.9}$$

Conditioning on the training data yields the GP posterior at test inputs $\boldsymbol{X}_*$:

$$p(f_* \mid \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{X}_*) = \mathcal{N}\big(\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*\big), \tag{4.10}$$

with posterior mean and covariance

$$\boldsymbol{\mu}_* = \boldsymbol{m}_* + K_{*X} \boldsymbol{A}^{-1} \big(\boldsymbol{y} - \boldsymbol{m}_X\big), \tag{4.11}$$
$$\boldsymbol{\Sigma}_* = K_{**} - K_{*X} \boldsymbol{A}^{-1} K_{X*}. \tag{4.12}$$

4.3 Method

Our closed-loop tactile ergodic control method consists of three parts: (i) surface preprocessing; (ii) tactile ergodic control; and (iii) robot control. The surface preprocessing computes the quantities that need to be calculated only once when the surface is captured. Tactile ergodic control generates the motion commands for the virtual coverage/exploration agent using the precomputed quantities from the surface preprocessing and the robot controller tracks the generated motion commands with a manipulator using impedance control. We focus on the surface preprocessing and tactile ergodic control in this chapter and refer the readers to the original publication [14] for the details of the robot control.

4.3.1 Surface Preprocessing for Tactile Ergodic Control and Mapping

For tactile robotics tasks our domain of interest are curved surfaces (i.e. 2-manifolds) and we measure the underlying manifold $\mathscr{M}$ as a point cloud $\mathscr{P}$ composed of $N_{\mathscr{P}}$ points using an RGB-D camera

$$\mathscr{P} := \left\{ (\boldsymbol{v}_i, \boldsymbol{c}_i) \;\middle|\; \begin{array}{l} \boldsymbol{v}_i \in \mathbb{R}^3,\ \boldsymbol{c}_i \in \{0,\dots,255\}^3 \\ \text{for } i = 1,\dots,N_{\mathscr{P}} \end{array} \right\}, \tag{4.13}$$

where $\boldsymbol{v}_i$ is the position of the i-th surface point in Euclidean space and $\boldsymbol{c}_i$ is the vector of RGB color intensities. For coverage tasks we assume there is a processing pipeline (i.e., such as [50] which processes the point cloud to obtain the target spatial distribution $g(\boldsymbol{v}_i, \boldsymbol{c}_i) = q_i$. In exploration and estimation tasks we consider the parameter that we are interested in is a discrete spatial distribution on the point cloud. Accordingly, in all cases tactile ergodic control uses a discrete spatial distribution $q(\boldsymbol{v}_i) = q_i$ on the point cloud $\mathscr{P}$ as its input.

We formulate a tactile ergodic controller that covers target spatial distributions on arbitrary surfaces. We propagate the information encoding the control objective by diffusing the source term similarly to HEDAC. However, we utilize the non-stationary ($\dot{u} \neq 0$) Riemannian diffusion equation allowing control over the desired smoothness by controlling the diffusion time τ

$$\dot{u}(\boldsymbol{x}, \tau) = \Delta_{\mathscr{M}} u(\boldsymbol{x}, \tau), \quad \boldsymbol{x} \in \mathscr{M} \tag{4.14}$$

The diffusion time τ is independent of the coverage time t used by the ergodic control algorithm. We set the initial condition $u(\boldsymbol{x}, 0)$ using the source term given in Eq. (2.27) which encodes the coverage objective at the t-th timestep of the coverage, i.e., $u(\boldsymbol{x}, 0) = \mu(\boldsymbol{x}, t)$. In order to solve Eq. (4.14) on irregular and discrete domains, such as point clouds, we discretize the problem in space and time using Eq. (2.13) and Eq. (2.15). We use $\boldsymbol{u}_\tau$ to refer to a vector encoding the whole field at the τ-th timestep indexed by the points on the point cloud $\mathscr{P}$.

Spectral Basis on Point Clouds

Note that solving Eq. (2.15) requires inverting a large sparse matrix, which might be computationally expensive depending on the size of the point cloud and requires the timestep τ to be set before the inversion. Alternatively, we can solve the problem in the spectral domain by projecting the original problem and reprojecting the solution back to the point cloud. This procedure generalizes using the Fourier transform for solving the diffusion equation on a rectangular domain in $\mathbb{R}^n$ to Riemannian manifolds. Note that the Fourier series are the eigenfunctions of the Laplacian Δ in $\mathbb{R}^n$. Therefore, we can use the eigenvectors of the discrete Laplacian $\boldsymbol{L}$ for solving the diffusion equation on point clouds.

We can write the generalized (i.e., $\boldsymbol{M} \neq \boldsymbol{I}$) eigenvalue problem for the Laplacian as

$$\boldsymbol{C}\boldsymbol{\phi}_m = \lambda_m \boldsymbol{M}\boldsymbol{\phi}_m, \tag{4.15}$$

where $\{\lambda_m, \boldsymbol{\phi}_m\}$ are the eigenvalue/eigenvector pairs. Since $\boldsymbol{M}$ is diagonal and $\boldsymbol{C}$ is symmetric positive definite, by the spectral theorem, we know that the eigenvalues are real, non-negative, and in ascending order analogous to the frequency. Therefore, we can use the first N_M eigenvalue/eigenvector pairs as a low-frequency approximation of the whole spectrum.

Our key insight is that the same basis can be used for both ergodic control and estimation on point cloud. Since the eigenvectors are orthogonal with respect to the inner product defined by $\boldsymbol{M}$, we can normalize them to obtain an orthonormal basis. In this orthonormal basis the diffusion PDE becomes a system of decoupled ordinary differential equations (ODEs) in the spectral domain, which can be solved efficiently for any diffusion time τ without the need for matrix inversion. Similarly, We can use the same orthonormal basis for constructing heat kernels in order to estimate spatial distributions on curved surfaces using Gaussian processes. We model the spatial distribution as an unknown scalar field defined on the surface, $f(\boldsymbol{x}) : \mathcal{M} \to \mathbb{R}$.

The distribution f can be either directly observable or inferred from measurements, for example given a viscoelastic estimation procedure such as the one in [10]. We can obtain pointwise estimates $y \approx f(\boldsymbol{x})$ at queried surface locations $\boldsymbol{x}$. We model the spatial distribution $f(\boldsymbol{x})$ using a GP and use the collected samples $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]^\top$ and corresponding estimates $\boldsymbol{y} = [y_1, \ldots, y_n]^\top$ to condition the posterior distribution using Eq. (4.10). Because the parameters spatially change on a curved surface, the GP covariance kernel $k(\boldsymbol{x}, \boldsymbol{x}')$ should respect the intrinsic geometry of the manifold $\mathcal{M}$. A principled method is to construct kernels from intrinsic differential operators on manifolds [16] such as the Laplace-Beltrami operator $\Delta_{\mathcal{M}}$.

$$k_\infty(\boldsymbol{x}, \boldsymbol{x}') = \frac{\sigma^2}{C_\infty} \sum_{m=0}^{\infty} \exp\left(-\frac{\kappa^2}{2}\lambda_m\right) \phi_n(\boldsymbol{x})\, \phi_m(\boldsymbol{x}'), \tag{4.16}$$

where C_∞ is a normalization constant and $\{(\lambda_m, \phi_m)\}_{n\geq 0}$ are the eigenpairs of the Laplace–Beltrami operator on $\mathcal{M}$. As we discussed in Section 2.2.1, on a Riemannian manifold the Laplace-Beltrami operator is defined only by the metric hence it is an intrinsic operator.

Therefore, it is preserved under isometric transformations. Note that the above heat kernel have closed-form expressions for various curved spaces [7, 6] but since we operate on point clouds, we need to construct a discrete kernel and work with the eigenvalues and eigenvectors of the discrete Laplacian $\boldsymbol{L}$ as given in Eq. (4.15).

Using the eigenvalues and eigenvectors, the discrete heat kernel on the point cloud becomes

$$k_{i,j} = \sigma^2 / C_\infty \sum_{k=0}^{n_k} e^{-\frac{l^2}{2}\lambda_k} \phi_{k,i}\phi_{k,j}, \tag{4.17}$$

where $C_\infty = \frac{\sum k_{i,i} M_{i,i}}{\sum M_{i,i}}$.

We assume the probed surface deforms isometrically, meaning that it can bend without stretching. Under this assumption, the intrinsic geometry of the surface remains unchanged, and thus the eigenvalues and eigenvectors of the Laplace-Beltrami operator are preserved. This property allows us to precompute the eigenpairs of the discrete Laplace–Beltrami given in Eq. (4.15) and the pairwise kernel values $k_{i,j}$ given in Eq. (4.17). Specifically, before the exploration phase, $k_{i,j}$ is evaluated for all pairs of points (i, j) in the reference point cloud, yielding a complete kernel lookup table that encodes the spatial correlations across the surface. During exploration, at each contact location along the trajectory, the point in the precomputed point cloud closest to the agent location is identified. The kernel matrix required for Gaussian Process regression is then assembled directly from the precomputed $k_{i,j}$ values, avoiding redundant kernel evaluations. This strategy enables faster estimation during exploration. Moreover, since the kernel values are precomputed, the number of eigenpairs employed only influences the offline precomputation time and does not affect the computational cost of the online kernel construction.

4.3.2 Tactile Ergodic Control

Our original method [14] is based on geometric algebra, which provides a powerful and elegant mathematical framework for representing and computing with geometric entities. However, it requires extensive background knowledge which is out of scope for this thesis. Therefore, we present our method using linear algebra.

We model the sensor used for exploration or the tool used for coverage as a *virtual agent*. The agent is fixed to the end-effector of a robotic manipulator and we denote its position in 3D Euclidean space as $\boldsymbol{x}_a$. Because the target surface is represented as a discrete point cloud $\mathscr{P}$, yet the agent moves continuously in $\mathbb{R}^3$, we project the virtual agent's position $\boldsymbol{x}_a$ onto the closest tangent plane as we will explain in the next section.

Projection to the Local Tangent Plane

A straightforward approach is to projecting the agent to the closest point on the point cloud. However, this can lead to discontinuities in the motion and we instead project the agent to the *local tangent space* of the surface. We also project the neighboring points to the local tangent space to later compute the gradient of the diffused field on the surface to guide our ergodic control agent.

- **Neighborhood Selection:** We query a K-D tree $\mathcal{T}(\mathcal{P})$ to find the K nearest points to $\boldsymbol{x}_a$. This set of points, $\mathcal{N}(a) = \{\boldsymbol{v}_j\}_{j=1}^{K}$, defines the local neighborhood.

- **Surface Fitting:** A tangent plane is fit to $\mathcal{N}(a)$ by minimizing the least-squares error of the plane equation. Given the estimated surface normal $\hat{\boldsymbol{n}}$ and a reference point $\boldsymbol{x}_r$ on the plane, we define the local tangent plane $T_{\boldsymbol{x}_r}\mathcal{P}$. We also use the local normal $\hat{\boldsymbol{n}}$ as a reference to align the orientation of the end-effector w.r.t. the surface.

- **Projection to Local Tangent Plane:** We project the agent and the local neighbors onto the local tangent plane using the estimated normal $\hat{\boldsymbol{n}}$

$$\boldsymbol{x}^{\parallel} = \boldsymbol{x} - \langle \boldsymbol{x} - \boldsymbol{x}_r, \hat{\boldsymbol{n}} \rangle \hat{\boldsymbol{n}}, \tag{4.18}$$

  Projected local neighborhood will be used for computing the gradient of the diffused field on the surface. Using the local orthonormal basis vectors $(\hat{\boldsymbol{t}}_1, \hat{\boldsymbol{t}}_2)$ of the tangent plane, the 2D coordinates $\boldsymbol{\xi}_j = (\xi_{j,1}, \xi_{j,2})^\top$ for each neighbor are derived:

$$\xi_{j,1} = \left\langle \boldsymbol{v}_j^{\parallel} - \boldsymbol{x}_a^{\parallel}, \hat{\boldsymbol{t}}_1 \right\rangle, \qquad \xi_{j,2} = \left\langle \boldsymbol{v}_j^{\parallel} - \boldsymbol{x}_a^{\parallel}, \hat{\boldsymbol{t}}_2 \right\rangle. \tag{4.19}$$

Coverage Computation

We need to compute the coverage of the agent on the target surface in order to set the initial condition for the diffusion equation. We represent the agent's footprint as a disk with radius r_a. This representation is versatile, as arbitrary tool or sensor footprints can be decomposed into a combination of multiple disks [13]. To update the coverage on the surface, we utilize a spatial kernel that maps the agent's presence onto the discrete point cloud $\mathcal{P}$. Let $\boldsymbol{p} \in \mathbb{R}^{|\mathcal{P}|}$ be a vector representing the cumulative coverage of the surface, where p_j is the coverage value associated with point $\boldsymbol{v}_j$.

Given the agent's current position $\boldsymbol{x}_a$ and the set of neighboring points $\mathcal{N}(a)$ computed at the previous step, we model the sensor footprint using a Gaussian radial basis function. The coverage contribution k_j for each neighbor is computed as:

$$k_j = \exp\left(-\frac{\|\boldsymbol{x}_a - \boldsymbol{v}_j\|^2}{r_a}\right). \tag{4.20}$$

The parameter r_a effectively scales the variance of the footprint, ensuring that points closer to the center of the virtual disk receive higher coverage intensity. The coverage state is then updated iteratively:

$$p_j^{(t+1)} = p_j^{(t)} + k_j, \quad \forall \boldsymbol{v}_j \in \mathcal{N}(a), \tag{4.21}$$

where t denotes the time step. Given the coverage vector $\boldsymbol{p}_t$ and the target distribution $\boldsymbol{q}$, we can compute the elements of the initial condition vector $\boldsymbol{u}_0$ for the diffusion equation using the source term given in Eq. (2.27). This formulation ensures that the initial condition reflects the remaining coverage requirement, with higher values indicating areas that require more attention from the agent.

Computing Diffused Field using the Spectral Basis

Due to the orthonormal transformation, the PDE on the point cloud becomes a system of decoupled ODEs in the spectral domain. The eigenvectors $\boldsymbol{\phi}_m$ are orthonormal with respect to the inner product defined by the mass matrix $\boldsymbol{M}$. Accordingly, we can stack the first N_M eigenvectors $\boldsymbol{\phi}_m$ as column vectors to construct the matrix $\boldsymbol{\Phi} \in \mathbb{R}^{N_{\mathcal{P}} \times N_M}$ encoding an orthonormal transformation $\boldsymbol{\Phi}^\top \boldsymbol{M} \boldsymbol{\Phi} = \boldsymbol{I}$. Then, we can transform the coordinates (shown with superscripts) from the point cloud to the spectral domain

$$\boldsymbol{u}^{\boldsymbol{\phi}} = \boldsymbol{\Phi}^\top \boldsymbol{M} \boldsymbol{u}^{\boldsymbol{x}}. \tag{4.22}$$

Note that this step is equivalent to computing the Fourier series coefficients of a target distribution in SMC. Due to the orthonormal transformation, the PDE on the point cloud becomes a system of decoupled ODEs in the spectral domain. It is well known that the solution of a first-order linear ODE $\dot{x}(\tau) = -cx(\tau)$ is given by $x(\tau) = e^{-c\tau}x(0)$, where c is a constant and $x(0)$ is the initial condition. Therefore, the solution of the system of ODEs in the spectral domain is given in matrix form as

$$\boldsymbol{u}_\tau^{\boldsymbol{\phi}} = \begin{bmatrix} e^{-\lambda_1 \tau} & \dots & e^{-\lambda_m \tau} \end{bmatrix}^\top \odot \boldsymbol{u}_0^{\boldsymbol{\phi}}, \tag{4.23}$$

where $\odot$ denotes the Hadamard product. We observe from Eq. (4.23) that the exponential terms with larger eigenvalues (i.e., higher frequencies) will decay faster. Therefore, approximating the diffusion using the first N_M components introduces minimal error. Secondly, similar to the mixed norm used in SMC, the low-frequency spatial features are prioritized. Next, we transform the solution back to the point cloud to get the diffused potential field

$$\boldsymbol{u}^{\boldsymbol{x}} = \boldsymbol{\Phi} \boldsymbol{u}^{\boldsymbol{\phi}}. \tag{4.24}$$

We can combine Equations (4.22), (4.23) and (4.24) into a unified spectral scheme

$$\boldsymbol{u}_\tau = \Phi \begin{bmatrix} e^{-\lambda_1 \tau} & \dots & e^{-\lambda_m \tau} \end{bmatrix}^\top \odot \left(\Phi^\top \boldsymbol{M} \boldsymbol{u}_0\right). \tag{4.25}$$

We omit the superscripts when working on the point cloud for brevity. Note that τ is the only free parameter in the diffusion computation. However, its value should be adapted according to the mean spacing between the adjacent points h on the point cloud. For that purpose, we introduce the hyperparameter $\beta > 0$ and embed it into the timestep calculation

$$\tau = \beta h^2. \tag{4.26}$$

Accordingly, we can control the diffusion behavior independently of the point cloud size. Increasing β results in longer diffusion times and attenuates the high-frequency spatial features. This corresponds to a more global coverage (see [13] for details). Conversely, decreasing β results in shorter diffusion times, which leads to preserving the high-frequency spatial features, hence more local coverage behavior.

Since the Laplacian is invariant to distance preserving (i.e., isometric) transformations such as rigid body motion or deformation without stretching we compute $\boldsymbol{C}$, $\boldsymbol{M}$ and derived quantities only once in the preprocessing step for a given surface. Recomputation is not necessary if the object stays still, moves rigidly, or the target distribution $\boldsymbol{q}$ changes. This enables efficient updates of the target distribution $\boldsymbol{q}$ using vision. For example, in a cleaning task, we can use a vision-based segmentation module to update $\boldsymbol{q}$ based on the current state of the surface. In exploration and estimation tasks, we can update $\boldsymbol{q}$ using the mean and covariance of the GP posterior. This allows us to adapt our coverage strategy in real-time as we gather more information about the surface.

Ergodic Control Command using the Gradient of the Diffused Field

As in HEDAC, we guide the coverage agent using the gradient of the diffused potential field as the acceleration command

$$\ddot{\boldsymbol{x}}_a(t) = \nabla u(\boldsymbol{x}_a^{\parallel}, \tau). \tag{4.27}$$

where $\nabla u(\boldsymbol{x}_a^{\parallel}, \tau)$ denotes the gradient of the diffused potential field at the projected agent position $\boldsymbol{x}_a^{\parallel}$. However, the projected agent is not necessarily located at a point of the point cloud, and the diffused field is only solved at the points of the point cloud. Therefore we use the values of the diffused field at the neighbor locations as the height $h_j = u_{j,\tau}$ of a second surface from the tangent plane. Then, we fit a 3-rd degree polynomial P to this surface as shown by using the weighted least squares objective

$$\hat{\boldsymbol{A}} = \arg\min_{\boldsymbol{A}} \operatorname{tr}\Big((\boldsymbol{y} - \boldsymbol{X}\boldsymbol{A})^\top \boldsymbol{W}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{A})\Big), \tag{4.28}$$

with the diagonal weight matrix $\boldsymbol{W} = \operatorname{diag}(w_1, w_2, \dots w_m)$. To prioritize the local accuracy near the agent, we introduce a diagonal weight matrix $\boldsymbol{W}$, where each entry w_j is determined by a Gaussian decay function:

$$w_j = \exp\left(-\frac{\|\boldsymbol{\xi}_j\|^2}{\max_k \|\boldsymbol{\xi}_k\|^2}\right). \tag{4.29}$$

Lastly, we calculate the gradient at the projected agent's position using the analytical gradients of the polynomial. We depict the approach visually in Figure 4.1. We refer readers to [74] for details of computing gradients on point clouds.

The partial derivatives of the fitted polynomial evaluated at a tangent plane point are computed analytically. We then lift the 2D tangent-plane gradient $\left(\frac{\partial P}{\partial \xi_1}, \frac{\partial P}{\partial \xi_2}\right)^\top$ back to the ambient space as

$$\nabla u(\boldsymbol{x}_a^{\parallel}, \tau), = \frac{\partial P}{\partial \xi_1}, \hat{\boldsymbol{t}}_1 + \frac{\partial P}{\partial \xi_2}, \hat{\boldsymbol{t}}_2 \tag{4.30}$$

This vector lies in the tangent plane $T_{\boldsymbol{x}_r}\mathscr{P}$ by construction, making it the intrinsic surface gradient of the diffused field u at the projected agent position $\boldsymbol{x}_a^{\parallel}$.

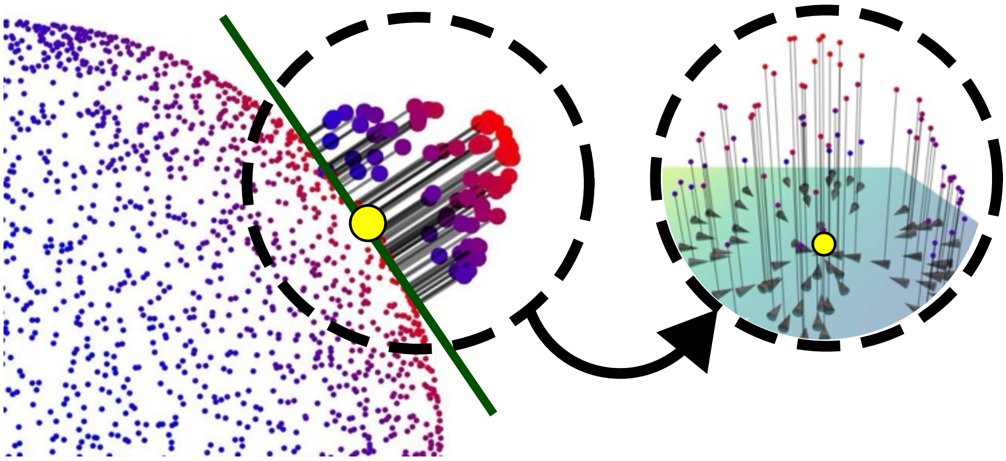


Figure 4.1: Blue-red points show the value of the potential field $\boldsymbol{u}_\tau$ on the pointcloud $\mathscr{P}$ and the yellow point is the projected agent position $\boldsymbol{x}_a^{\parallel}$. We also project the local neighborhood $\mathscr{N}(a)$ to the tangent plane $T_{\boldsymbol{x}_r}\mathscr{P}$, shown in green. Next, we use the height function $h_j = u_{j,\tau}$ which uses the values of the potential field to lift the projected points in the normal direction of the tangent plane. We show the lifted points with large blue-red points. We fit a polynomial to this lifted surface and compute its analytical gradients at the neighbor locations $\nabla u(\xi_j, \tau)$, as shown with arrows in the detail view.

4.3.3 Target Spatial Distribution

Target exploration and coverage uses the same ergodic controller but differs on how we set the target distribution $\boldsymbol{q}$ on the surface.

- **Tactile Coverage**: we set the target distribution $\boldsymbol{q}$ by processing the measured point cloud. For example, in a cleaning task a segmentation module can set q_i to be proportional to the deviation from the object surface color to encode the dirty regions.

- **Tactile Exploration and Estimation**: we set the target distribution $\boldsymbol{q}_t$ using the mean and covariance of the GP posterior and ergodic exploration generate samples for conditioning the GP in a closed-loop manner. Notably in this setting the target distribution can change over time. As the target distribution for the ergodic controller, we use an expected information density (EID) $\boldsymbol{q}_t$ where individual elements are computed as

$$q_{i,t} = \alpha \frac{\max(\mu_i - \bar{\mu}, 0)}{\sum_{i=1}^{N} \max(\mu_i - \bar{\mu}, 0)} + (1-\alpha) \frac{\sigma_i}{\sum_{i=1}^{N} \sigma_i}. \tag{4.31}$$

A weighting parameter $\alpha \in [0,1]$ balances exploitation of regions with high estimated stiffness and exploration of regions with high uncertainty by combining the posterior mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma}$ of the GP.

4.4 Experiments

We conducted simulation experiments to evaluate the performance of our method and comparing it against the baselines. In real-world experiments we aim to demonstrate the practical applicability of our approach on a physical robotic system.

4.4.1 Simulation Experiments

In simulation experiments, we aim to address the following questions: Q1: Role of intrinsic surface geometry and topology in exploration and mapping. Q2: Effect of the exploration coefficient. Q3: Effect of decoupling exploration and estimation updates. Q4: Effect of system–dynamics on exploration. Q5: Ergodic exploration with different spectral weighting strategies. During simulation experiments, we utilized three distinct point clouds, each representing a unique set of challenges:

- **Stanford Bunny:** Sourced from the Stanford 3D Scanning Repository [1], we utilized a partial point cloud of the bunny. This choice serves as a proxy for real-world scenarios where robots operate with incomplete environmental models due to partial views. Its complex geometry and irregular external boundaries allow us to evaluate the method's ability to handle intricate surfaces.

- **Plate with Multi-modal Distribution:** We collected a point cloud of a plate with complex, multi-modal hand-drawn distribution using an Intel RealSense RGB-D camera. This object was selected specifically to test the method's performance against complex, multi-modal distributions that arise during tasks such as cleaning.

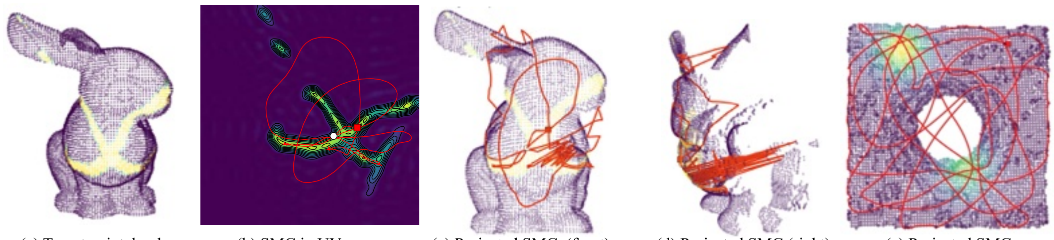
- **Deformable Silicon Sample:** Also captured via a RealSense camera, this sample presents a non-trivial topology (an internal hole). This allows us to evaluate how our method adapts to shape changes and objects with complex connectivity.

[1] https://graphics.stanford.edu/data/3Dscanrep/

Intrinsic Surface Geometry and Topology (Q1)

First baseline that we consider is a 2D parameterization-based ergodic control approach, which we refer to as UV-ergodic control. This is a straightforward extension of the standard 2D ergodic control algorithms to point clouds.

In this approach, the point cloud is parameterized using a simple UV mapping technique for point clouds based on PCA, and then apply a standard 2D ergodic control algorithm SMC [68] in the UV space. The resulting trajectory is then mapped back to the point cloud using the inverse UV mapping. We show the resulting trajectories using UV-ergodic-control in Figure 4.2. We show how our method performs with the same objects in Figures 4.3 and 4.4.



(a) Target pointcloud (b) SMC in UV space (c) Projected SMC (front) (d) Projected SMC (right) (e) Projected SMC

Figure 4.2: Distortion introduced by UV parameterization in ergodic control. (a) Point cloud with an X-shaped information target highlighted in yellow. (b) Target and exploration trajectory in UV space, with the trajectory shown in red. (c,d) Front and right views of the point cloud showing the SMC trajectory mapped from UV space back onto the surface. (e) Additional example illustrating a failure case of the UV parameterized SMC approach on a target object with an internal hole.

Figure 4.2 shows that for objects with complex geometry and boundaries, the parameterization introduces distortion and tearing, which leads to discontinuous motion on the surface. In addition, internal boundaries are not represented in the planar domain (see Figure 4.2(e)), so the exploration can incorrectly traverse regions that should remain separated. In contrast, our method (see Figures Figure 4.4 and Figure 4.3) preserves the intrinsic surface structure and naturally respects both external and internal boundaries, resulting in smooth and consistent exploration trajectories.

Effect of Exploration Coefficient on Estimation Performance (Q2)

Secondly, we analyzed the role of the exploration coefficient α, which regulates the trade-off between exploration and exploitation during mapping (see Eq. (4.31)). We show the effect of α on the search behavior qualitatively in Figure 4.3.

For low α, the agent prioritizes exploration and traverses informative regions without dwelling, continuing the search for other salient areas (see Fig. 4.3, first row). As α increases, the time spent within relevant regions increases accordingly. For sufficiently large values, once the agent rapidly encounters the first informative region, it predominantly exploits its local

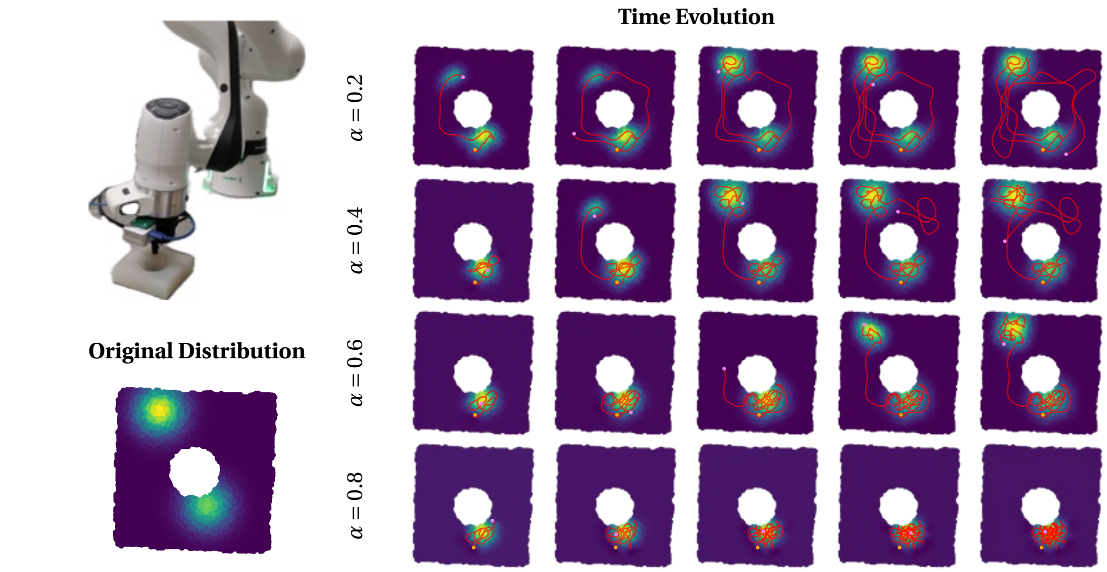


Figure 4.3: Comparison of the time evolution of the ergodic search for different values of the exploration coefficient α. (top left) Gaussian Dimple phantom used to generate the pointcloud; (bottom left) underlying target distribution; (right) search progress over time displayed at uniformly spaced time intervals. Each successive column corresponds to the same fixed simulation time increment, and all sequences are shown up to a final time of 150 s. The plots include the updated distribution estimate and the agent's trajectory (red). Top view of the sample; the dimple is truncated for visualization, appearing as an interior hole.

neighborhood; this behavior is clearly visible in the bottom-right panels of Figure 4.3 for $\alpha = 0.8$, where, within the fixed simulation horizon, the second informative region remains unexplored. Moderate values of α lead to a good balance between exploration and exploitation as expected.

In order to quantify the effect of α we compared the performance of different values in terms of the root mean square error (RMSE) between the estimated distribution and the ground truth distribution

$$\mathrm{RMSE}(t) = \sqrt{\frac{1}{N_{\mathscr{P}}} \left\| \boldsymbol{q}_t - \boldsymbol{q}_{\mathrm{GT}} \right\|_2^2}, \tag{4.32}$$

where $\boldsymbol{q}_t$ is the estimated distribution at time t. $\boldsymbol{q}_{\mathrm{GT}}$ is the ground truth and $N_{\mathscr{P}}$ is the number of points of the point cloud. For each value of α, 50 independent simulations were conducted, and the mean and standard deviation of each metric were computed. To ensure comparable performance across all runs, the simulation was terminated when the reconstruction error satisfied the stopping criterion RMSE < 2 kPa, indicating that the prescribed target accuracy had been achieved, or when a maximum duration of 400 s was reached, whichever occurred first. We show the estimation accuracy (RMSE) and convergence time for different values of α in Table 4.1.

We selected $\alpha = 0.6$ for the rest of the experiments as it provides a good balance between

Table 4.1: Effect of α on ergodicity, reconstruction accuracy (RMSE), and convergence time. Values are reported as mean ± std over 50 runs. Simulations stop when RMSE < 2 or after 400 s if convergence is not achieved.

α	Ergodic Metric	RMSE []	Time [s]
0.0	0.094 ± 0.009	2.02 ± 0.28	344 ± 42
0.1	0.112 ± 0.012	1.91 ± 0.09	243 ± 51
0.2	0.141 ± 0.036	1.90 ± 0.10	136 ± 55
0.3	0.148 ± 0.028	1.93 ± 0.04	112 ± 29
0.4	0.145 ± 0.019	1.89 ± 0.07	108 ± 16
0.5	0.134 ± 0.014	1.92 ± 0.06	134 ± 17
0.6	0.124 ± 0.019	1.93 ± 0.06	182 ± 27
0.7	0.115 ± 0.017	1.98 ± 0.02	249 ± 32
0.8	0.115 ± 0.026	3.14 ± 1.92	384 ± 21

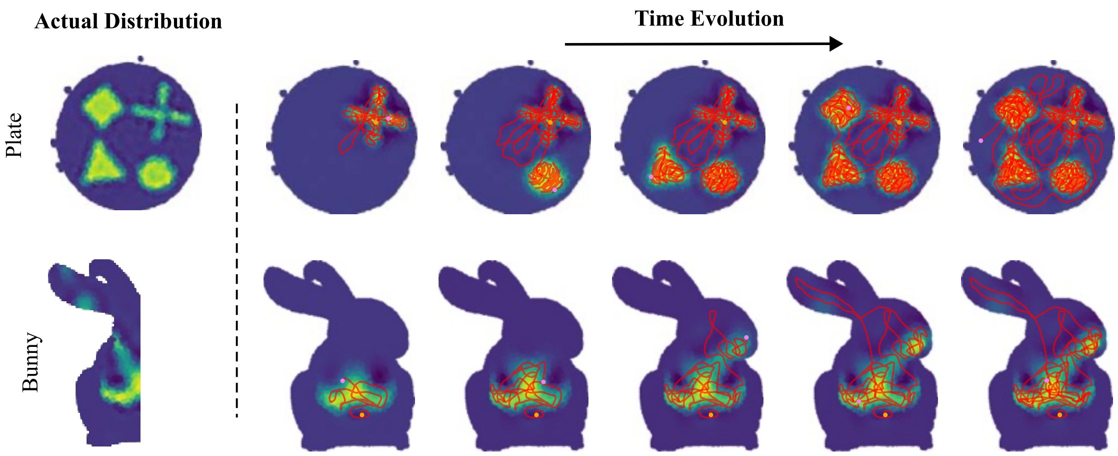


Figure 4.4: Time evolution of the ergodic search with a fixed exploration coefficient $\alpha = 0.6$ for different point-cloud shapes. The left column shows the underlying target distributions, while the right columns illustrate the temporal evolution of the search at uniformly spaced time intervals, including updates of the estimated distribution and the agent's trajectory (red).

exploration and exploitation, resulting in faster convergence to the target distribution while maintaining a reasonable level of ergodicity. We show the time evolution of the search for $\alpha = 0.6$ in Figure 4.4. It can be seen that the search is able to discover all modes of the distribution as well as their shapes and details correctly.

Update Interval (Q3)

The update interval determines how frequently the estimated distribution is updated using newly acquired measurements and therefore directly affects the responsiveness of the search strategy. Ergodic control allows decoupling the update interval for exploration and estimation. This is a key advantage as it allows to balance the computational load and the performance. Figure 4.5 reports the distribution of the ergodic metric and the reconstruction error (RMSE) at the end of the exploration time for increasing update intervals over multiple runs. The update interval denotes the number of samples collected before the Gaussian Process model is updated via regression with the newly acquired data.

For small update intervals (high-frequency updates), both the ergodic metric and the RMSE remain consistently low, indicating that frequent updates enable the agent to closely match the target distribution while maintaining an accurate estimate of the underlying stiffness field. In this regime, newly acquired information is rapidly incorporated, resulting in stable convergence and limited variability across trials. As the update interval increases (lower frequency updates), performance progressively degrades. Beyond 100, both metrics exhibit a marked increase in median value and dispersion. Large update intervals delay the incorporation of measurements, causing the agent to rely on outdated estimates and leading to suboptimal coverage of salient regions. This behavior is reflected in the sharp increase of the ergodic metric as well as in the substantial growth of the reconstruction error.

Also these results highlight a trade-off between computational efficiency and performance. While larger update intervals reduce computational overhead, they significantly degrade both exploration quality and estimation accuracy. Figure 4.5 suggests there is a critical threshold between 20 and 100 for the update interval, beyond which performance deteriorates rapidly. Accordingly, in the following experiments, the update interval is selected as 50 samples to preserve estimation fidelity while avoiding unnecessary computational load.

System Dynamics aware Exploration (Q4)

Standard BO typically ignores robot motion during the sampling decision and requires frequent, computationally expensive Gaussian Process (GP) updates to remain effective. To ensure a fair "matched budget" comparison, we implemented a local BO variant that utilizes the same surface-aware GP and batch-update strategy as PC-HEDAC. In this setup, the agent collects samples continuously along its trajectory in batches of 50, matching the update interval used in our method. As shown in Figure 4.6, even with matched computational and distance budgets, ergodic control approaches consistently outperform BO baselines. This gap highlights the benefit of system-dynamics-aware exploration (Q4): while BO selects points in a greedy fashion, then enforcing constraints; ergodic controllers generates continuous, smooth trajectories that naturally account for the robot's movement constraints.

Ergodic Exploration with Different Spectral Weighting Strategies (Q5)

We further compared PC-HEDAC against two Laplacian-based SMC spectral weighting schemes: (i) polynomial weighting $(1+\sqrt{\lambda_k})^{-2}$ and (ii) exponential weighting $\exp(-\gamma\sqrt{\lambda_k})$. All methods shared the same Laplacian operator and GP hyperparameters, with each controller tuned to its best-performing exploration-exploitation coefficient α. The results in Figure 4.6 indicate that PC-HEDAC achieves a lower RMSE than SMC variants. We attribute this to a fundamental lim-

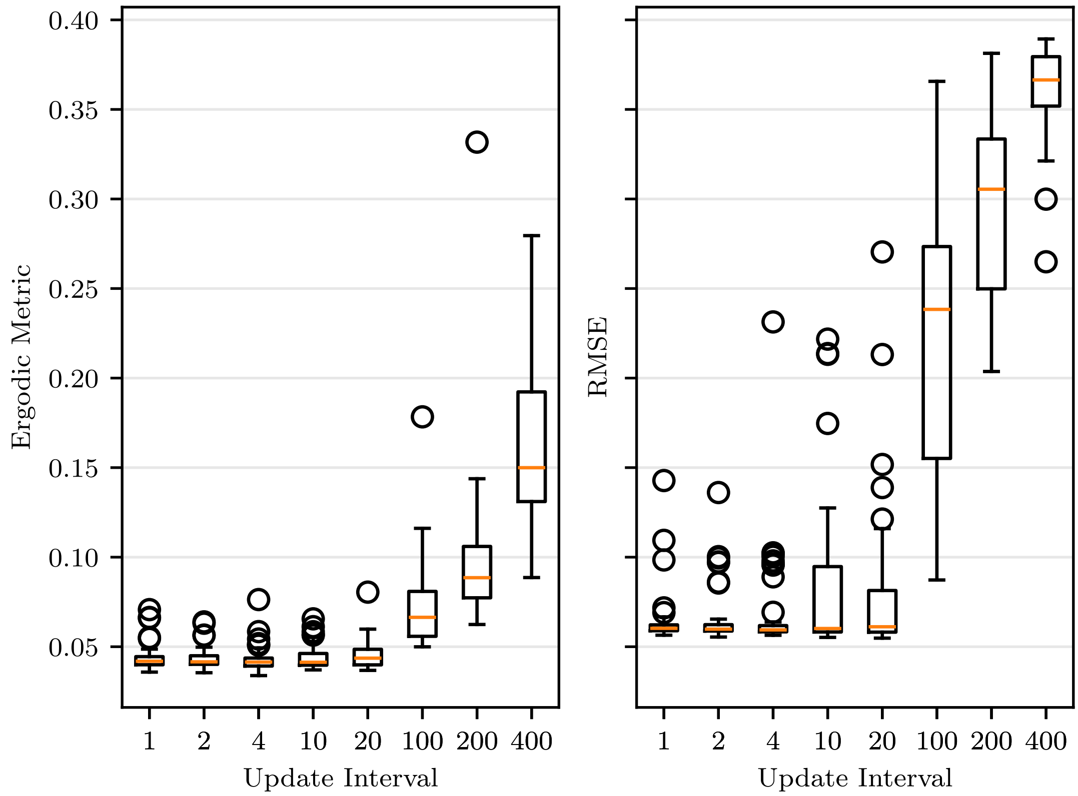


Figure 4.5: Effect of the update interval on performance. Boxplots show the ergodic metric (left) and reconstruction error (RMSE, right) for increasing update intervals over multiple runs. Larger intervals result in degraded performance and increased variability.

itation of SMC-style controllers: while they are efficient at discovering disjoint modes (global exploration), they often underrepresent fine-scale geometry within a single mode due to the attenuation of high-frequency components during basis truncation. In contrast, diffusion-based approach provides a more interpretable mechanism for tuning exploration scales (Q5). Through the parameter β, which determines the diffusion time, we can set the minimum spatial feature scale of interest. From a heat kernel perspective, β directly corresponds to the GP lengthscale parameter l, effectively linking the resolution of the robotic planner to the fidelity of the underlying estimation model. This interdependent structure ensures that the smooth geometric representations used for control are mathematically consistent with the representations used for estimation.

4.4.2 Real-world Experiments

We performed real-world experiments both for the coverage and exploration/estimation settings. In the coverage setting, we demonstrate the method on a cleaning task where the robot cleans an arbitrary dirty region on a previously unknown curved surface (see Fig. 4.7).

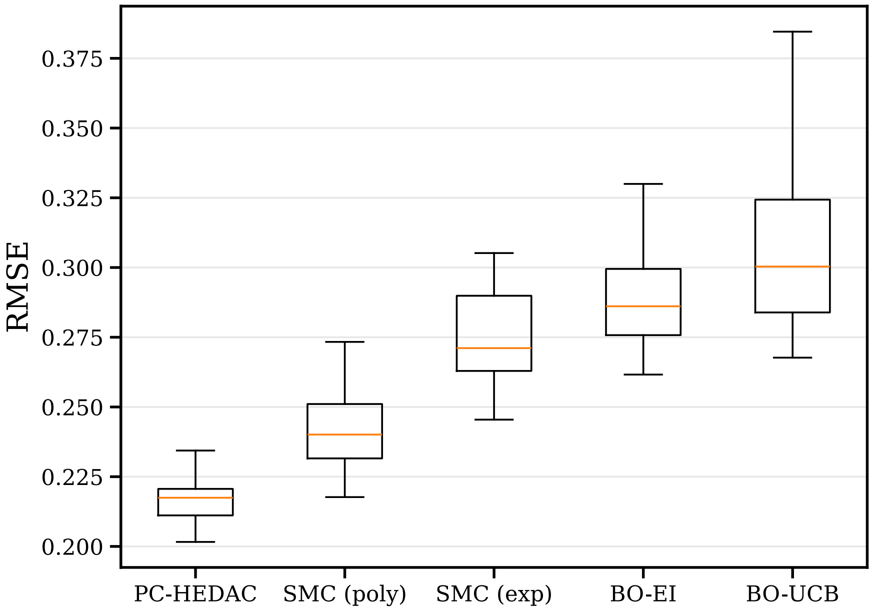


Figure 4.6: Comparison of our method to BO and ergodic control baselines in terms of the RMSE using ground truth.

In the exploration/estimation setting, we demonstrate the method on a palpation task where the robot needs to explore a curved deformable surface to estimate its viscoelastic properties and to detect inclusions (see Fig. 4.8).

In both of these tasks our experimental setup comprises a BotaSys SensOne 6-axis force torque (F/T) sensor and Intel Realsense camera attached to the wrist of a 7-axis Franka robot manipulator and a 3-D printed part attached to the F/T sensor. The custom part interfaces a sponge at its tip for cleaning experiments and a probe for palpation experiments. We consider the tool center point to be the coverage agent's position.

The output of the tactile ergodic controller is the gradient of the diffused field at the projected agent position. We use this gradient as the acceleration command for a simulated second-order system with unit mass and damping to compute the target position. As the target orientation we use the surface normal at the projected agent position. By combining the target position and orientation, we define a line target for the geometric task-space impedance controller. A line target has a nullspace along the line direction, which allows the robot to apply a force along the line while following the position and orientation targets in the other directions. The controller has been implemented using the open-source geometric algebra for robotics library *gafro*[2] that is presented in [65].

[2] https://gitlab.com/gafro

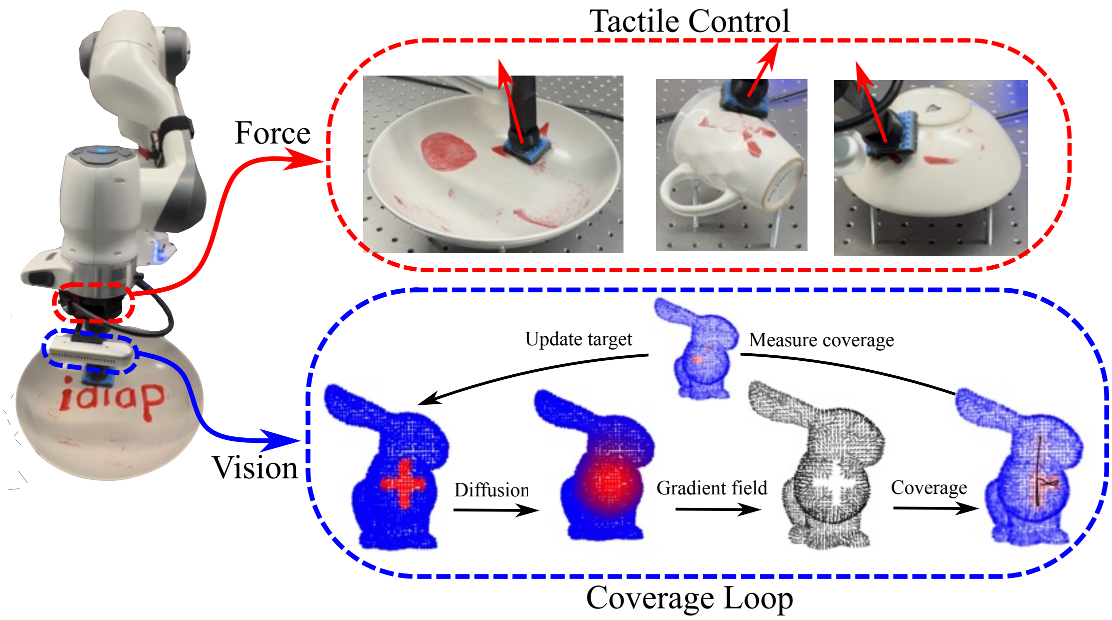


Figure 4.7: Overview of our feedback control method for tactile coverage. *Left:* We measure the surface and the red target using the camera and encode them in a point cloud. *Bottom-right:* We diffuse the target and use its gradient field to guide the coverage. Then, we close the loop by measuring the actual coverage with the camera and use it as the next target. *Top-right:* We measure the tactile interaction forces using the force sensor and the tool orientation using the joint positions. We solve the geometric task-space impedance control problem using a line target and a force target along the line.

Cleaning Experiments

To validate our method, we conducted real-world cleaning experiments using three distinct kitchen utensils: a plate, a bowl, and a cup. These objects were selected to provide a variety of curved geometries and were marked with different target distributions, including geometric shapes, the “RLI” logo, and an “X” mark. For stability during physical interaction, the objects were fixed to the workspace. At the start of each trial, the robot moved to a predefined joint configuration to capture the target distribution via a wrist-mounted RGB-D camera. A key challenge incorporated into these experiments was the use of a single image frame for surface acquisition. This resulted in partial and noisy point clouds, providing a realistic testbed for the robustness of our smooth geometric representations against the “imperfect sensing” described in the abstract. We summarize the visual results of these experiments in Figure 4.9 and provide comprehensive recorded data and videos on the accompanying website.

As illustrated in Figure 4.9, our method demonstrates real-time adaptation to both surface geometry and complex target distributions. This adaptability effectively guides the robot across the manifold to achieve precise coverage of the specified regions. Notably, the method remains resilient to sensor noise and partial observations, which is a critical requirement when operating on the incomplete point clouds acquired during the experiments.

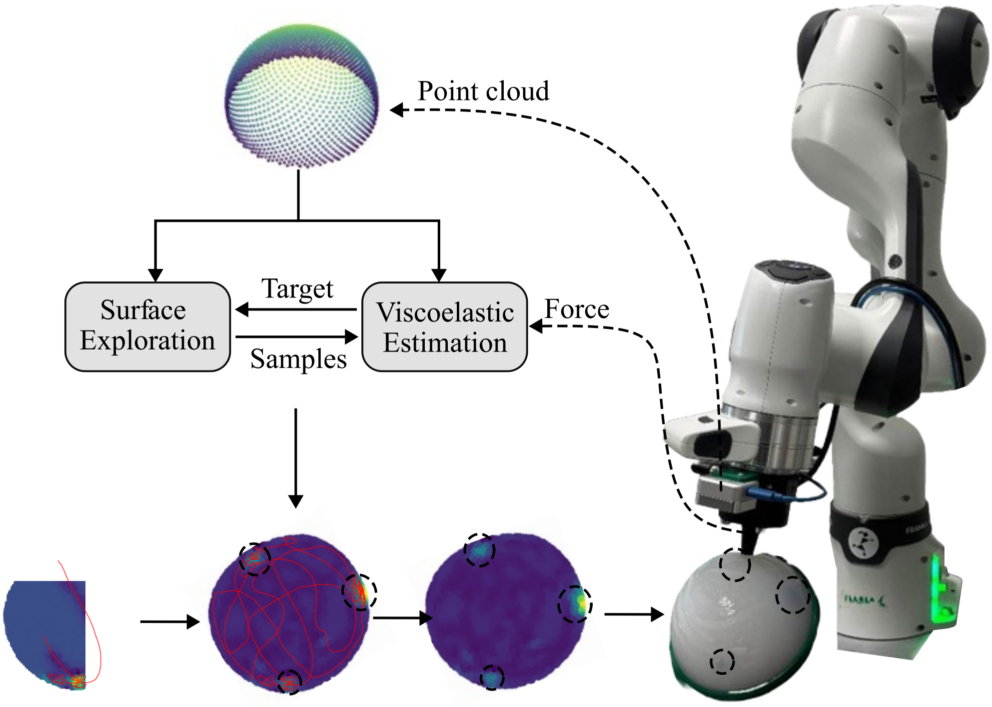


Figure 4.8: Method overview. (*top left*) The surface is sampled online as a point cloud and processed to construct a surface-aware basis for exploration and viscoelastic estimation. Exploration collects samples for viscoelasticity estimation, while the planner returns target regions for subsequent exploration. (*bottom left*) Evolution of the estimated viscoelastic map during palpation. The red curve shows the exploration trajectory, and the yellow regions inside dashed circles indicate inclusions detected by our method. (*right*) Snapshot from the experiments of a robotic arm performing autonomous palpation on a hemisphere phantom.

The high computational efficiency of our formulation enables closed-loop control, allowing the system to respond dynamically to unmodeled task dynamics and changing environmental conditions. For instance, if a specific region is not cleaned properly due to complex physical interactions, the controller detects the residual target and automatically adjusts its trajectory to prioritize that area. Ultimately, by bridging intrinsic geometry with spatial smoothness through diffusion processes, we establish an online regularizer that ensures stable, real-time performance even in the presence of significant sensing uncertainty.

A primary challenge in tactile coverage tasks remains the *measurability* of progress; in scenarios such as surface sanding, mechanical palpation, or high-precision inspection, progress is not always directly observable via an RGB-D camera. While we utilized cleaning as a proxy task to validate our method, this reveals a dependency on visual feedback to close the loop. However, this limitation suggests a novel human-robot interaction modality: a human expert can use easy-to-remove markers to annotate regions of interest on a complex surface, which the robot can then autonomously process and cover. This collaborative approach leverages

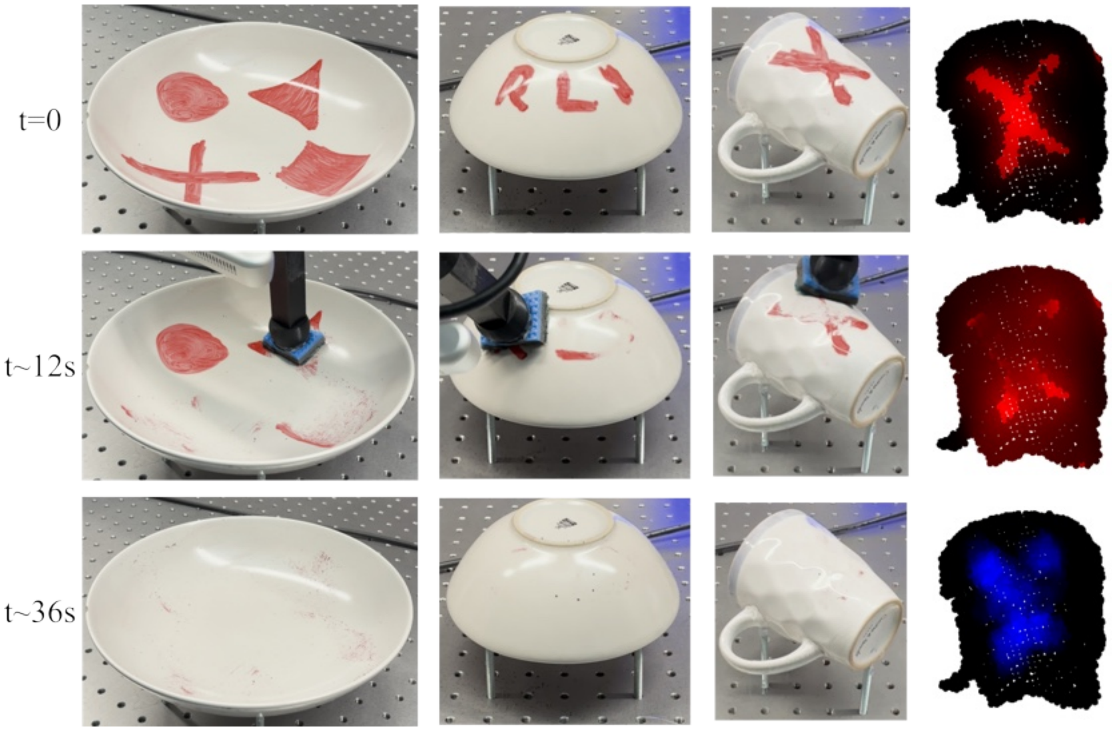


Figure 4.9: Real-world experiment of the robot cleaning a plate, a bowl, and a cup. For the first three columns we give snapshots from the initial, intermediate, and final states from top to the bottom. In the last column, we show the target distribution $\boldsymbol{q}$, the simulated potential field $\boldsymbol{u}_\tau$ and the coverage $\boldsymbol{p}_t$ from top to the bottom.

expert knowledge while maintaining the robotic system's geometric autonomy.

Viscoelastic Mapping Experiments

Two custom silicone phantoms with distinct geometries, boundaries, and viscoelastic properties were fabricated to evaluate the proposed algorithm. We refer the reader to the original paper [9] for detailed information on the phantom design and fabrication process.

The first phantom *hemisphere phantom* was designed to replicate the geometry and physical characteristics of the human breast. Three spherical inclusions were embedded within the phantom: two with a radius of 5 mm and one with a radius of 1 cm.

The second phantom *Gaussian Dimple phantom* incorporates a flat, stiff inclusion of size 1.5x1.5 cm, a spherical inclusion with a radius of 5 mm, and a small U-shaped lesion (2x5 mm) and corresponds to the deformable silicon sample used in the simulated experiments.

For each silicone sample, eight independent palpation experiments were performed under different initial exploration conditions. Because no physical ground-truth elasticity distribution is available, an empirical reference was constructed from the asymptotic behavior of the viscoelasticity reconstruction algorithm. To this end, exploration was allowed to continue well beyond the nominal convergence time, such that the final reconstructed viscoelasticity maps of each experiment could be regarded as a steady-state estimate of the underlying tissue properties on the point cloud. Convergence was verified by monitoring the RMSE variation over time, which remained below a predefined tolerance before extracting the reference map. We denote the final reconstructed elasticity map of the experiment k with the vector $\boldsymbol{q}_k(t_f)$ where the i-th element of this vector is the final stiffness of the i-th vertex of the point cloud $\mathscr{P}$. The empirical ground truth was then defined as the cross-experiment consensus of the final maps.

To obtain an unbiased estimate of viscoelasticity map reconstruction accuracy, a leave-one-out strategy was adopted. For each experiment k, the reference map was computed as the mean of the final maps obtained from all remaining experiments,

$$\boldsymbol{q}_{\mathrm{GT}}^{(-k)} = \frac{1}{N-1} \sum_{j=1,\ j\neq k}^{N} \boldsymbol{q}_j(t_f) \tag{4.33}$$

where $N = 8$ denotes the number of experiments performed on the same sample. The reconstruction error over time was then quantified using the following modified RMSE formula,

$$\mathrm{RMSE}_k(t) = \sqrt{\frac{1}{N_{\mathscr{P}}} \left\| \boldsymbol{q}_k(t) - \boldsymbol{q}_{\mathrm{GT}}^{(-k)} \right\|_2^2}, \tag{4.34}$$

where $N_{\mathscr{P}}$ denotes the number of points in the point cloud $\mathscr{P}$. Finally, exploration was stopped when the variation of the ergodic metric over a sliding window of 20 s became smaller than 10^{-3} in magnitude and the metric value dropped below 0.3. Under this condition, the coverage discrepancy exhibits no further significant reduction, indicating convergence to steady-state behavior. This stopping rule enables a quantitative assessment of the exploration time required to reach asymptotic reconstruction accuracy.

The quantitative results of the experiment are reported in Table 4.2 and they confirm that reconstruction accuracy improves with increasing stiffness contrast and region size, while smaller or weaker anomalies are more challenging to detect. Although the Dimple phantom exhibits a higher absolute elasticity RMSE (2.26 kPa) due to its larger baseline elasticity (125 kPa versus 35 kPa for the hemisphere), the hemispherical sample shows a higher relative error (approximately 3.1% versus 1.8%). This difference is explained by the sharper stiffness peaks and larger stiffness contrast in the hemispherical sample (greater than 2× compared to approximately 1.4×), which make local reconstruction more challenging despite the lower overall stiffness. Conversely, exploration time is primarily governed by the spatial extent of the domain: the smaller Dimple phantom reaches convergence significantly faster, highlighting the dependence of convergence rate on surface size. Importantly, accurate reconstruction is achieved in both cases despite substantial differences in geometry and mechanical properties, demonstrating that the proposed exploration and estimation method generalizes across shapes and elasticity regimes. In addition, elasticity peaks are found in all experiments, demon-

Table 4.2: Reconstruction accuracy and stopping time across palpation experiments. Values are reported as mean ± standard deviation. Subscript el stands for elasticity and vi for viscosity.

Silicone sample	$\mathrm{RMSE_{el}}$ [kPa]	$\mathrm{RMSE_{vi}}$ [Pas]	t_{stop} [s]
Hemisphere	1.07 ± 0.17	2.47 ± 0.67	370 ± 23
Gaussian Dimple	2.26 ± 0.38	1.85 ± 0.28	186 ± 42

strating consistent detection of stiffness maxima and robustness of the method to variations in surface geometry and material contrast. Figure 4.10 shows the qualitative behavior of the exploration over time in one experiment per phantom. As expected, the agent spends more time in high-stiff regions and only passes through region with small, constant elasticity.

Similar to tactile coverage experiments, the real-world experiments demonstrate robustness to sensing uncertainty and real-time adaptability to complex surface geometries and material properties. Moreover, using the same smooth geometric representations for both control and estimation ensures that the exploration strategy is inherently aligned with the estimation model and vice-versa.

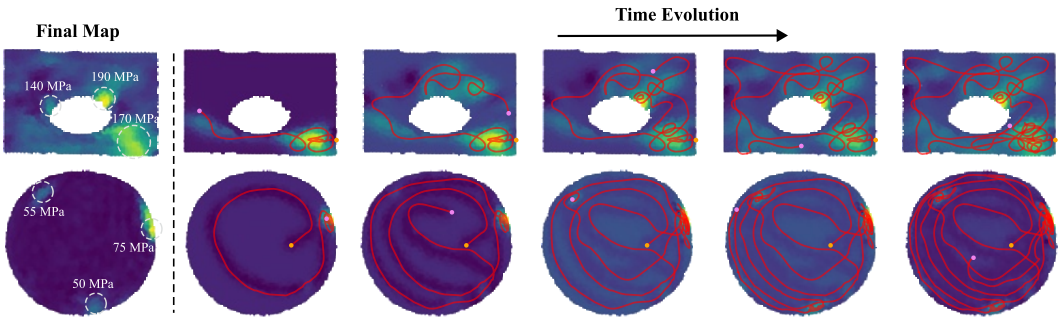


Figure 4.10: Temporal evolution of the reconstructed viscoelastic map during ergodic palpation of the Gaussian dimple and hemisphere phantoms. From left to right: reference distribution, progressive reconstruction, and exploration trajectory. The method accurately localizes stiff regions and converges to a stable estimate, consistent with the quantitative results in Table 4.2. Example explorations and online map reconstructions are available also in the multimedia attachment.

4.5 Conclusion

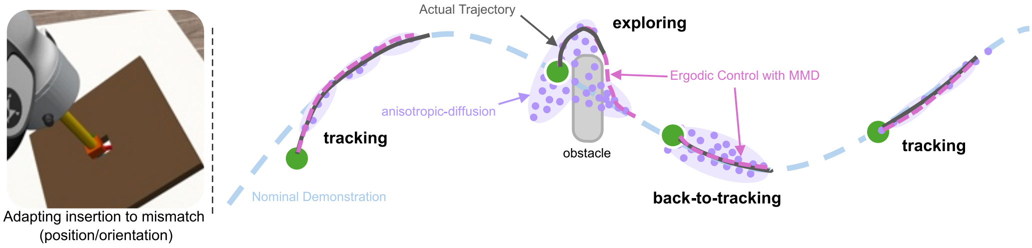
We used smooth geometric representations induced by Laplacian operator to develop a closed-loop ergodic control method for tactile coverage and exploration tasks on curved surfaces. Our method directly operates on point clouds, enabling real-time adaptation to complex geometries and target distributions while maintaining robustness to sensing uncertainty. Through simulated and real-world experiments, we demonstrated the effectiveness of our approach in achieving robust coverage and viscoelastic mapping. By bridging intrinsic geometry with spatial smoothness through diffusion processes, we established a method that can adapt and generalize to different object shapes and distributions on the fly.

While this chapter focuses on tactile tasks on curved surfaces, the proposed method is fundamentally generalizable to any domain where a suitable Laplacian operator or heat kernel can be defined. A compelling future direction involves applying the heat kernel to the manifold of Symmetric Positive Definite (SPD) matrices, enabling the autonomous exploration and estimation of variable stiffness gains. Furthermore, extending this method to the product manifold $\mathbb{R}^3 \times SO(3)$ would allow for structured exploration and estimation within the full 6D pose space. By leveraging the intrinsic geometry of these manifolds, the same diffusion-based regularizers could provide a unified approach for a wide range of robotic learning and control problems, from active learning of variable impedance to active learning of pose distribution, while maintaining the advantages demonstrated in this chapter.

4.5.1 Limitations & Future Work

We presented a reactive control method that does not require trajectory optimization or long-term planning. While this allows for real-time performance and robustness to unmodeled dynamics, it also limits the ability to plan for long-term objectives or constraints. Existing ergodic trajectory optimization methods are not directly applicable to HEDAC as we do not have an explicit ergodic metric that we can optimize. However, recent advances in flow matching based ergodic coverage [94] enables using a reference vector field instead of an explicit metric. Therefore one can use the vector field generated by HEDAC as the reference for flow matching and optimize trajectories that follow this vector field.

5 Adaptive Ergodic Imitation for Exploration around Demonstrations



In this chapter, we present an adaptive exploration and tracking framework that combines diffusion-based target generation and ergodic control with imitation learning (IL) for enhancing runtime adaptation. We construct an adaptive target distribution by anisotropically diffusing the distribution of IL action trajectories based on the local action geometry and estimated phase lag. This target serves as the input for a receding horizon ergodic controller. We evaluate the framework in simulation and qualitatively compare it against standard imitation learning and ergodic control methods.

Publication Note

The material presented in this chapter is adapted from the following workshop paper.

- Xu, Z.*, **Bilaloglu, C.***, Li, Y. and Calinon, S. (2026). Ergodic Imitation for Adaptive Exploration around Demonstrations. *Workshop on Geometry in the Age of Data-Driven Robotics (ICRA)*

Z. Xu was supervised by C. Bilaloglu for his course project and helped C. Bilaloglu with the conceptualization, implementation, experiments, visualization, and writing.

5.1 Introduction

IL is a practical paradigm for robot programming, where the intended behavior is learned by demonstrations without an explicit reward. In practice, however, the objective is rarely to replicate a demonstration exactly; rather, the agent must adapt the observed behaviors to previously unseen environments that might diverge from the training distribution. Recent critiques of deep generative models in robotics suggest that these architectures often overfit the demonstration data, essentially memorizing specific action sequences rather than learning generalizable policies [37, 57]. Consequently, current IL approaches remain notoriously brittle even under minimal distribution shifts. This necessitates a shift toward imitation paradigms that prioritize situational adaptation over pure trajectory replay.

Although recent adaptive exploration methods guide systems to sample different modes from the dataset, they explore in the latent space [76, 48]. Consequently, they lack the continuous exploration capabilities needed for the subtle, state-space adjustments typical of tasks like insertion, assembly or articulation. We employ a different perspective that treats demonstrations as references to be tracked under nominal conditions, by using them as informed priors for exploration when environmental shifts render pure tracking insufficient. Instead of considering tracking and exploring as two different objectives, we use the ergodic control methodology to formulate tracking as a special case of exploration. This results in a continuous behavioral spectrum, from rigid imitation to adaptive exploration, governed by the same underlying controller.

Two previous works have extended ergodic control to the IL setting, applying it to insertion [87, 93], cart-pole inversion, and surface cleaning [52] tasks. In this imitation context, demonstrations are leveraged to construct a static target distribution that the ergodic controller uses for task reproduction. A primary advantage of ergodic imitation is using state-visitation statistics rather than strict temporal sequencing; this allows the robot to synthesize successful behaviors without being tethered to the demonstrator's specific timeline [52]. While this flexibility is beneficial for tasks like cleaning or insertion, temporal dependency remains critical for most manipulation tasks. To address this, we propose an adaptive formulation using local geometry of the action trajectories to strike a balance between these two paradigms. Our approach reproduces the temporal characteristics of the demonstration when the environment permits, yet autonomously transitions to ergodic exploration if the robot becomes "stuck" and can not track the reference. Our contributions can be summarized as follows:

- formulating a framework for adapting to environmental variations by combining imitation learning, ergodic control, and adaptive target distribution generation;
- geometry-aware anisotropic diffusion for synthesizing goal-oriented target distributions inducing adaptive tracking and exploration behaviors;
- modulating the diffusion strength for anisotropic diffusion using an estimated phase lag between the system and the expert context, which serves as a proxy for task progress;
- formulating ergodic controller and anisotropic diffusion using the heat kernel in SE(3) based on the product of SO(3) and $\mathbb{R}^3$ heat kernels.

5.2 Background

Maximum mean discrepancy (MMD) provides a discrepancy measure for ergodic control [40], particularly effective when the target distribution is represented using a set of discrete samples $\{\boldsymbol{q}_i\}_{i=1}^{N} \subset \mathbb{R}^3$. For a trajectory represented by discrete points $\boldsymbol{x}_t$, the squared MMD between the coverage $p_{\boldsymbol{x}}$ and the target distribution q is approximated as:

$$\overline{\mathrm{MMD}}_k^2(p, q) = \frac{1}{T^2} \sum_{t=0}^{T-1} \sum_{t'=0}^{T-1} k(\boldsymbol{x}_t, \boldsymbol{x}_{t'}) - \frac{2}{TN} \sum_{t=0}^{T-1} \sum_{i=1}^{N} k(\boldsymbol{x}_t, \boldsymbol{q}_i) + z(q), \tag{5.1}$$

where $z(q) = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} k(\boldsymbol{q}_i, \boldsymbol{q}_j)$ is a constant term that depends solely on the target distribution.

5.3 Method

We treat a pre-trained imitation learning policy (e.g., Diffusion Policy [23] or retrieval-based methods [57]) as a black-box sampler of observation conditioned action trajectories (chunks) $\{\boldsymbol{x}_{t+k}\}_{k=0}^{T_i}$ where each individual action $\boldsymbol{x}_{t+k} \in \mathrm{SE}(3)$ represents the target pose of the robot's end-effector, and T_i is the horizon. We obtain the input for our method, by querying the policy to draw N independent samples from this conditional distribution in parallel

$$\mathcal{S}_t = \left\{ \left\{ \boldsymbol{x}_{t+k}^{(i)} \right\}_{k=0}^{T_i} \right\}_{i=1}^{N}.$$

The proposed framework is agnostic to the specific imitation learning method and how the action chunks are generated. For simplicity we present our method using a retrieval-based imitation learning approach similar to [57]. Consequently, the IL formulation simplifies to retrieving the N-nearest observations to the current state from the demonstration dataset $\mathcal{D}$ and utilizing their corresponding action trajectories as input to our framework.

The full adaptive ergodic imitation pipeline composes three core components: phase-lag estimation, target distribution generation, and receding-horizon ergodic trajectory optimization as shown in Figure 5.1.

5.3.1 Phase-Lag Estimation

We evaluate the current task-progress relative to the demonstration by estimating a phase lag $e_p(t)$ between the current execution and the expert context

$$e_p(t) = s^*(t) - \hat{s}(t) \tag{5.2}$$

where $\hat{s}(t)$ is the estimated phase and $s^*(t)$ is the target phase. The estimated phase $\hat{s}(t)$ is the temporal index of the retrieved observation from demonstration dataset to the current

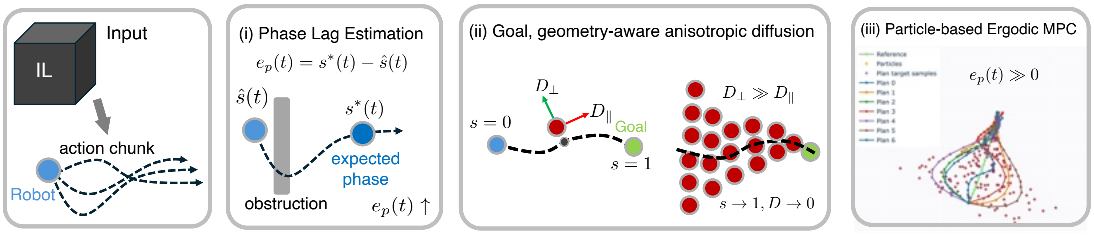


Figure 5.1: Overview of the adaptive ergodic imitation pipeline. As its input the system queries the pre-trained imitation learning policy to obtain N candidate action trajectories. (i) Phase lag estimation module provides a signal to modulate (ii) the anisotropic diffusion for target distribution generation. (iii) The resulting target distribution is fed into a receding horizon ergodic controller that optimizes the trajectory to balance tracking and exploration.

observation. In contrast, the target phase is an adaptive target based on system's task progress:

- Progressing $\rightarrow |e(t)| \leq \epsilon$, the robot is tracking the expert context. We advance the target phase ($\dot{\tau} = 1$) and reset the stagnation counter $c = 0$.
- Stagnating $\rightarrow |e(t)| > \epsilon$, the robot is lagging behind the expert context. We stop the target phase ($\dot{\tau} = 0$) and increment the stagnation counter c.

The stagnation signal c provides a measure of task progress and determines whether to track or explore around demonstrations by modulating anisotropic diffusion. To maintain focus on the overarching framework rather than individual component engineering, we utilize this straightforward metric. Nonetheless, more sophisticated task-progress and failure detection methods could be integrated, such as recent anomaly monitoring techniques [36] or vision-language-based robotic evaluation models [58].

5.3.2 Target Distribution Generation

Retrieval or diffusion policy-based imitation learning methods only provide samples $\boldsymbol{x}_t^{(i)}$ from the underlying distribution p_t. Therefore we represent the target distribution for the ergodic controller by processsing discrete samples. Accordingly we use a stochastic differential equation (SDE) formulation working directly with particles $\boldsymbol{q}_j$ instead of a PDE-based formulation. After retrieving N state trajectories we use anisotropic diffusion to generate a particle-based distribution $\mathcal{Q}_t = \{\boldsymbol{q}_j\}_{j=1}^{M} \subset \mathrm{SE}(3)$ using the following SDE:

$$\mathrm{d}\boldsymbol{q}_j = \underbrace{\beta(\theta)\,\nabla_{\boldsymbol{q}_j} \log p_t(\boldsymbol{q}_j)\,\mathrm{d}\tau}_{\text{score-based drift-term}} \quad + \quad \underbrace{\sqrt{2\,\Sigma(\theta, \boldsymbol{q}_j)}}_{\text{anisotropic diffusion}} \quad \mathrm{d}\boldsymbol{W}_\tau, \tag{5.3}$$

where τ is the diffusion time step and $\boldsymbol{W}_\tau$ is a standard Wiener process, and $\theta(t) \in [0,1]$ is diffusion modulator. Diffusion modulator is a temperature-like variable that modulates the balance between tracking and exploration using the stagnation counter c and kernelized coverage ρ (introduced in next subsection):

$$\dot{\theta} = f(c(t), \rho(t)), \quad \theta(0) = 0. \tag{5.4}$$

Diffusion modulator starts at zero and increases when the agent is stagnating and the coverage is low. Conversely, it decreases when the agent is progressing towards the target phase. For small θ, the score-based drift term dominates and leads to tracking behavior; as θ increases, anisotropic diffusion becomes more prominent and lead to exploration. We detail how to compute and the role of each term of SDE in the following subsections.

Heat-kernel score

The drift term biases particles toward regions of high reference density, helping them stay close to the reference distribution during the tracking phase. It is kernel-weighted average of the individual kernel scores:

$$\nabla_{\boldsymbol{q}_j} \log p_t(\boldsymbol{q}_j) = \frac{\sum_{i=0}^{N} \sum_{k=0}^{T_i} k_\tau(\boldsymbol{q}_j, \boldsymbol{x}_{t+k}^{(i)}) \nabla_{\boldsymbol{q}_j} \log k_\tau(\boldsymbol{q}_j, \boldsymbol{x}_{t+k}^{(i)})}{\sum_{i=0}^{N} \sum_{k=0}^{T_i} k_\tau(\boldsymbol{q}_j, \boldsymbol{x}_{t+k}^{(i)})}, \tag{5.5}$$

where k_τ is the heat kernel on SE(3) implemented as a product of a translational heat kernel on $\mathbb{R}^3$ and a rotational heat kernel on SO(3). While a product kernel $k_\tau^{\mathbb{R}^3} \times k_\tau^{\mathrm{SO}(3)}$ technically treats the pose space as the direct product $\mathbb{R}^3 \times \mathrm{SO}(3)$ rather than the Special Euclidean group SE(3), it serves as an efficient and structurally sound approximation for real-time receding horizon control. Because our task coordinate frame remains fixed during execution, the loss of left-invariance does not affect convergence, while the decoupled structure preserves the computational efficiency for online operation.

For two poses $\boldsymbol{T}_1, \boldsymbol{T}_2 \in \mathrm{SE}(3)$, we compute the relative transform

$$\boldsymbol{T}_{12} = \boldsymbol{T}_1^{-1} \boldsymbol{T}_2,$$

from which we extract the relative translation $y \in \mathbb{R}^3$ and the relative rotation angle $\phi \in [0, \pi]$. The full kernel is written as

$$k_\tau^{\mathrm{SE}(3)}(\boldsymbol{T}_1, \boldsymbol{T}_2) = k_\tau^{\mathbb{R}^3}(\boldsymbol{y})\, k_\tau^{\mathrm{SO}(3)}(\phi).$$

The translational component is the standard heat kernel on $\mathbb{R}^3$ given in Eq. (2.18) The rotational component is evaluated using a truncated spectral expansion of the SO(3) heat kernel,

$$k_\tau^{\mathrm{SO}(3)}(\phi) = \sum_{\ell=0}^{\ell_{\max}} (2\ell + 1) \exp(-\ell(\ell+1) D_r \tau)\, \chi_\ell(\phi),$$

where D_r is the rotational diffusion coefficient and χ_ℓ is the character of the ℓ-th irreducible representation of SO(3),

$$\chi_\ell(\phi) = \frac{\sin\big((2\ell+1)\phi/2\big)}{\sin\big(\phi/2\big)}.$$

Near $\phi = 0$, we use the limiting value

$$\chi_\ell(0) = 2\ell + 1$$

to avoid numerical instability. The spectral expansion is truncated at order $\ell_{\max}$, and the coefficients

$$a_\ell = (2\ell+1)\exp(-\ell(\ell+1)D_r\tau)$$

are precomputed. For scale consistency, the rotational kernel can be normalized by its value at the identity,

$$k_\tau^{\mathrm{SO}(3)}(0) = \sum_{\ell=0}^{\ell_{\max}} (2\ell+1)^2 \exp(-\ell(\ell+1)D_r\tau).$$

Anisotropic diffusion

To promote goal-oriented exploration around the expert context we use anisotropic diffusion leveraging time-dependency and geometry of the trajectories. More specifically, we use higher diffusion coefficients in the beginning of the trajectories compared to the end and higher diffusion in local normal direction compared to the local tangent. Let $\hat{\boldsymbol{t}}$ denote the local unit tangent at the projected particle $\boldsymbol{q}_j$ to the action chunk (see Fig. 5.1 (ii)). We split the diffusion term using:

$$\begin{aligned}\sqrt{2\,\Sigma(\theta,\boldsymbol{q}_j)}\,d\boldsymbol{W}_\tau = {} & \sqrt{2D_\parallel(\theta)}\left(\hat{\boldsymbol{t}}\hat{\boldsymbol{t}}^\top\right)d\boldsymbol{W}_\tau \\ & + \sqrt{2D_\perp(\theta)}\left(\mathbf{I}-\hat{\boldsymbol{t}}\hat{\boldsymbol{t}}^\top\right)d\boldsymbol{W}_\tau,\end{aligned} \tag{5.6}$$

where $D_\parallel$ and $D_\perp$ are the tangential and orthogonal diffusion coefficients, respectively. Choosing $D_\perp \gg D_\parallel$ promotes exploration primarily in directions normal to the trajectory while preserving coherence along it. We visualize the anisotropic diffusion in Fig. 5.2.

5.3.3 Particle-based Receding Horizon Ergodic Control

Given the anisotropically diffused target distribution represented by particles $\{\boldsymbol{q}_j\}_{j=1}^M \subset \mathrm{SE}(3)$, we address the problem of spatial exploration by optimizing a sequence of control inputs such that the resulting robot trajectory matches the target distribution. We quantify the discrepancy between the planned trajectory and this target distribution using the Maximum Mean Discrepancy (MMD) metric within a reproducing kernel Hilbert space (RKHS) similar to Hughes *et al.* [40]. However, instead of representing an object surface using the particle distribution here we represent the anisotropically diffused action chunks.

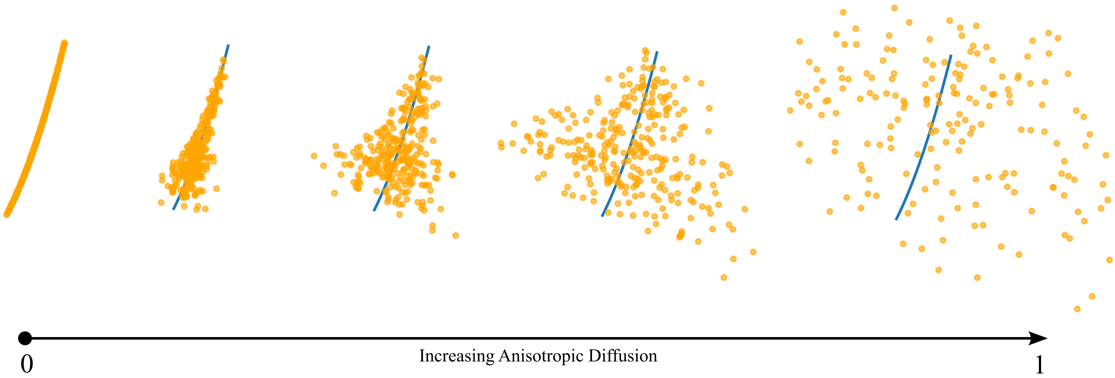


Figure 5.2: Effect of increasing anisotropic diffusion modulator θ on the resulting particle distribution. Blue is the reference trajectory and orange are the resulting particles.

System Dynamics and Horizon Parameterization

Let $H_p \in \mathbb{N}$ denote the finite planning horizon. At any execution time step t, the predicted robot state sequence over the look-ahead window is represented by the sequence $\mathscr{X}_t = \{\mathbf{X}_{t+h}\}_{h=0}^{H_p}$, where each state is a homogeneous transformation matrix living on the Special Euclidean manifold:

$$\mathbf{X}_{t+h} \in \mathrm{SE}(3) \subset \mathbb{R}^{4\times 4} \tag{5.7}$$

The decision variable sequence to be optimized at time step t is the command history $\mathscr{U}_t = \{\mathbf{u}_{t+h}\}_{h=0}^{H_p-1}$. Here, each control vector $\mathbf{u}_{t+h}$ is a spatial twist vector representing combined linear and angular velocities:

$$\mathbf{u}_{t+h} = \begin{bmatrix} \boldsymbol{v}_{t+h} \\ \boldsymbol{\omega}_{t+h} \end{bmatrix} \in \mathbb{R}^6 \cong \mathfrak{se}(3) \tag{5.8}$$

which maps directly to the associated Lie algebra $\mathfrak{se}(3)$ via the standard wedge operator $(\cdot)^\wedge : \mathbb{R}^6 \to \mathfrak{se}(3)$:

$$\mathbf{u}_{t+h}^\wedge = \begin{bmatrix} \boldsymbol{\omega}_{t+h}^\wedge & \boldsymbol{v}_{t+h} \\ \mathbf{0}^T & 0 \end{bmatrix} \tag{5.9}$$

where $\boldsymbol{\omega}^\wedge \in \mathfrak{so}(3)$ is the skew-symmetric matrix mapping the cross product operation for angular velocities. To structurally preserve the geometric constraints of SE(3) the system is governed by a discrete-time first-order geometric integrator model expressed in the end-effector frame:

$$\mathbf{X}_{t+h+1} = \mathbf{X}_{t+h} \exp\left(\mathbf{u}_{t+h}^\wedge \Delta t\right), \quad h \in \{0, \ldots, H_p - 1\} \tag{5.10}$$

where $\Delta t > 0$ is the discretization time step, and $\exp(\cdot) : \mathfrak{se}(3) \to \mathrm{SE}(3)$ denotes the matrix exponential map.

Ergodic Objective Function

The trajectory optimization problem is formulated as minimizing a composite cost function consisting of the squared MMD coverage metric and a regularizing control effort penalty:

$$\min_{\mathcal{U}_t} \mathrm{MMD}^2\big(\mathcal{X}_t(\mathcal{U}_t) \cup \mathcal{M}_t, \mathcal{Q}_t\big) + w_{\text{ctrl}} \sum_{h=0}^{H_p-1} \|\mathbf{u}_{t+h}\|_2^2 \tag{5.11}$$

where $w_{\text{ctrl}} > 0$ is a weighting scalar, and $\mathcal{M}_t$ denotes the coverage memory containing previously executed trajectory segments stored in a ring buffer with length T_m. The squared MMD measures the distance between the distribution of the planned trajectory $\mathcal{X}_t$ concatenated with the history $\mathcal{M}_t$ and the anisotropically diffused target particles $\mathcal{Q}_t$. Ignoring normalization factor and terms constant with respect to T, the finite-sample objective becomes:

$$\begin{aligned}
\mathrm{MMD}^2\big(\mathcal{X}_t(\mathcal{U}_t) \cup \mathcal{M}_t, \mathcal{Q}_t\big) = {} & \underbrace{\frac{1}{H_p^2} \sum_{h=1}^{H_p} \sum_{k=1}^{H_p} k_\tau(\mathbf{X}_{t+h}, \mathbf{X}_{t+k})}_{\text{Self-Repulsion}} - \underbrace{\frac{2}{H_p M} \sum_{h=1}^{H_p} \sum_{j=1}^{M} k_\tau(\mathbf{X}_{t+h}, \mathbf{q}_j)}_{\text{Target Attraction}} \\
& + \underbrace{\frac{2}{H_p T_m} \sum_{h=1}^{H_p} \sum_{l=1}^{T_m} k_\tau(\mathbf{X}_{t+h}, \mathbf{M}_{t+l})}_{\text{Historical Novelty Seek}} - \underbrace{\frac{2}{T_m M} \sum_{l=1}^{T_m} \sum_{j=1}^{M} k_\tau(\mathbf{M}_{t+l}, \mathbf{q}_j)}_{\text{Coverage}} \\
& + \underbrace{C(\mathcal{M}_t, \mathcal{Q}_t)}_{\text{Constant Term}}
\end{aligned} \tag{5.12}$$

The analytical expansion of the objective function reveals the underlying mechanics driving exploration:

- Self-Repulsion: Penalizes self-overlapping sequences within the current prediction window, forcing the planned trajectory to maximize its directional spread.
- Target Attraction: Minimizes the distance between the state sequence and the SDE target particles $\mathcal{Q}_t$, dragging the agent toward dense regions of the target distribution.
- Repulsion from Past Coverage: Penalizes the agent for revisiting regions recorded in the coverage memory $\mathcal{M}_t$. This explicit memory dependency prevents local trapping and continuously enforces long-horizon exploration across environmental variations.
- Coverage: A term that measures the similarity between the historical coverage and the target distribution we denote it with ρ. We use this term to modulate θ together with the stagnation counter c to ensure we do not expand the target distribution unless it is already well covered by the history.
- Constant: Represents terms that are not functions of controls

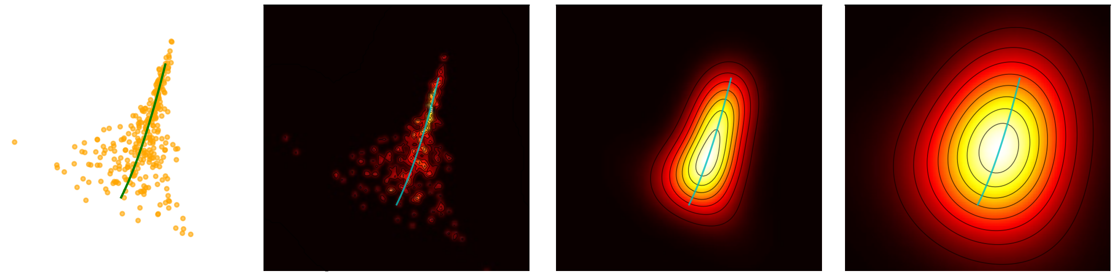

(a) Particle Distribution $\mathcal{Q}_t$ (b) $\tau = 0.001$ (c) $\tau = 0.01$ (d) $\tau = 0.1$

Figure 5.3: Effect of changing heat kernel's diffusion time τ on the resulting target distribution. (a) Input particle distribution, (b)-(d) heat kernel smoothed distribution with increasing diffusion time τ.

5.4 Experiments

We performed ablation studies to evaluate the individual contributions of each component of the framework. Following these characterizations, the full framework was validated in a 2D maze environment which requires both tracking and exploration. Finally, we demonstrated the method on a peg-in-hole insertion task in SE(3), introducing spatial and angular misalignment between the nominal demonstrations and the test-time environment to show scalability.

5.4.1 Ablation Studies

MMD-based ergodic controller perceives the target distribution through the lens of a kernel. Therefore, we investigated how the heat kernel interacts with the anisotropically diffused particle distribution, and how the kernelized target distribution can induce different exploration behaviors. First we investigated the diffusion time parameter τ of the heat kernel, which controls the smoothing effect of the kernel. This parameter is crucial for ensuring that the target distribution is sufficiently smooth for the ergodic controller, while still retaining enough geometric structure from the original demonstrations.

As shown in Fig. 5.3, a small τ does not provide sufficient smoothing, resulting in a discontinuous target distribution which in turn leads to erratic exploration behavior. Conversely, a large τ over-smooths the distribution, washing out important features of the task. Empirically, an intermediate scale of $\tau = 0.01$ satisfies this trade-off, generating a continuous potential field that ensures stable receding-horizon execution while capturing the underlying structure of the demonstrated task. Although the optimal τ may vary across different tasks and environments, we found that this value provides a good starting point, and can be further tuned based on the specific requirements of the task at hand. Beyond spatial regularization, an additional advantage of heat-kernel smoothing is that it substantially reduces the particle cardinality required to approximate the target distribution effectively. As shown in Fig. 5.4, when utilizing an optimal diffusion scale of $\tau = 0.01$, the structural profile of the reconstructed target distribution remains remarkably stable across varying particle counts. We observed that a population

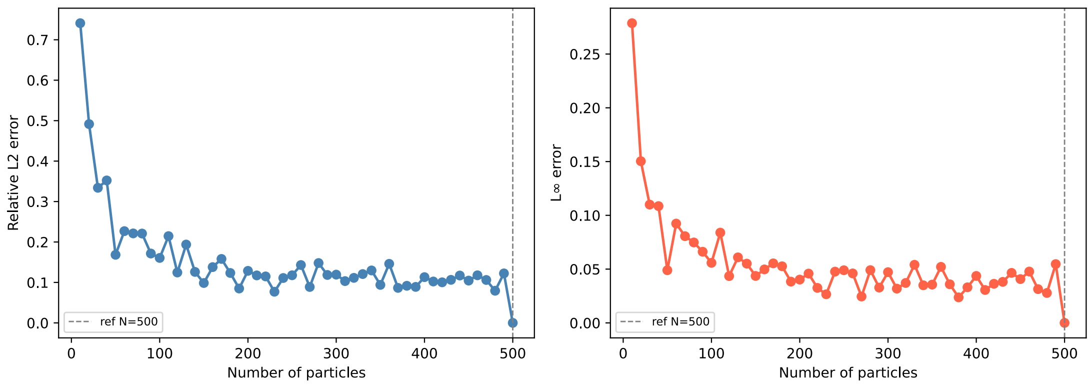


Figure 5.4: Effect of number of particles on the resulting heat kernel smoothed distribution with $\tau = 0.01$.

size of $M = 200$ particles satisfies the trade-off between computational efficiency and statistical approximation quality. This sample size minimizes the online evaluation footprint of the receding-horizon control loop while preserving the geometric fidelity necessary to represent the underlying task features. Next, to verify whether modulating the anisotropic diffusion can continuously interpolate between exploitation (tracking) and exploration regimes, we evaluated the complete pipeline against an isolated reference trajectory while systematically scaling the diffusion modulator θ. As demonstrated in Fig. 5.5, under a conservative modulation scale (small θ), the target distribution remains tightly localized around the reference trajectory, constraining the controller to a strict, minimal-variance tracking behavior. As θ is incrementally increased, the target distribution undergoes a continuous spatial broadening along the permitted manifold directions, smoothly transitioning the agent into an exploratory regime centered around the nominal path. At the upper limit (large θ), the target distribution becomes significantly delocalized, generating a relaxed potential field that empowers the controller to execute extensive spatial exploration and deviate substantially from the original demonstration to chart unvisited regions of the workspace. In a more realistic setting, the demonstrations may be multimodal, and the agent may be initialized in between two nominal trajectories. In such case, the anisotropic diffusion should still be able to induce exploration along both modes. As shown in Fig. 5.6 both anisotropic diffusion and heat kernel smoothing can be applied to multimodal demonstrations, resulting in a target distribution that captures the structure of both modes while still being smooth enough for effective tracking. We also show the resulting plans from the ergodic controller for a unimodal and multimodal scenario in Fig. 5.7. In the unimodal case, the plans are concentrated around the single reference trajectory, while in the multimodal case, the plans are more spread out, covering both modes of the target distribution.

In a generalized deployment setting, the action chunks may exhibit a multimodal structure. Under these conditions, the anisotropic diffusion and heat-kernel smoothing must remain capable of simultaneously preserving the structure of and inducing exploration along both modes.

As shown in Fig. 5.6, the combined pipeline of anisotropic diffusion and heat-kernel smoothing scales naturally to multimodal demonstration sets. This joint regularization yields a coherent target distribution that faithfully preserves the underlying dual-mode topological structure while maintaining the smoothness. This preservation of multiple modes is mathematically intuitive: since diffusion processes smooth their local profiles without merging distinct, far-apart modes into a single falsified average.

We further characterize this behavior by contrasting the resulting receding-horizon plans generated by the ergodic controller across unimodal and multimodal scenarios in Fig. 5.7. In the unimodal regime, the optimized paths remain tightly concentrated around the singular reference trajectory. Conversely, in the multimodal regime, the generated plans smoothly bifurcate and broaden, systematically distributing control authority to cover both modes of the underlying target distribution without requiring manual behavior tuning or discrete mode-switching heuristics.

5.4.2 Simulation Experiments

We evaluated the proposed framework in a 2D maze environment featuring narrow vertical apertures separated by a lateral offset. This environment was selected because it is conceptually minimal, yet uniquely suited to expose the fundamental failure modes of baseline imitation learning and standard ergodic control paradigms.

In this scenario, standard imitation learning approaches fail because the physical displacement of the gap introduces a severe distribution shift that the policy cannot generalize to out-of-the-box. Concurrently, standard ergodic control approaches fail because they lack the trajectory-tracking mechanisms necessary to drive the agent sequentially from the left to the right side of the maze.

To validate our method's adaptation capabilities, demonstrations were first collected within a nominal maze layout. At test time, the vertical gap locations were physically shifted to induce a deployment mismatch. Under this perturbation, the agent cannot simply replay the nominal trajectory and must instead adaptively explore its local surroundings to discover the new opening as shown in Fig. 5.8.

When the agent's forward path is blocked by a wall, the tracking discrepancy automatically triggers the anisotropic diffusion mechanism, which broadens the target distribution. This dynamic regularisation transforms the local potential field, driving the ergodic controller to initiate a targeted spatial exploration sequence that successfully guides the agent through the relocated aperture.

To evaluate our method quantitatively, we tested 50 distinct aperture configurations by introducing spatial offsets relative to the nominal opening position. These perturbations were sampled from a Gaussian distribution with $\sigma = 0.05$. We quantify the environmental mismatch using the gap displacement, which is defined as the normalized distance between the test-time aperture center and the nominal demonstration center. For example, if the nominal gap is centered at 50% of the total workspace height and the perturbed gap is displaced to 25%, the resulting gap displacement is calculated as 25%/50% = 50%.

A trial is classified as a success if the agent navigates through the maze to reach the terminal goal within a maximum budget of 1000 discrete control steps, utilizing the identical nominal trajectory across all test cases. In this highly constrained, obstacle-blocked scenario, pure retrieval-based or generative imitation learning approaches would yield zero success due to the out-of-distribution nature of the perturbed gap positions.

In contrast, as shown in Fig. 5.9, our framework leverages exploration, to find a feasible path through the relocated aperture. Although the success rate decreases and the average completion time increases as the gap displacement grows, the method maintains a robust performance even under significant perturbations. Notably, the main limiting factor for larger gap displacements is the fixed time budget and memory capacity.

To assess the viability of the framework in curved spaces, we implemented the system for insertion task in SE(3) in MuJoCo, confirming functional execution (Fig. 5.10).

5.5 Conclusion

In this chapter, we presented an adaptive ergodic imitation framework that bridges imitation learning and receding-horizon ergodic control through adaptive target generation. Our focus was on integrating different modules into a coherent framework for adaptive exploration rather than methodological contributions to individual components. We hope this would pave the way for utilizing ergodic control in novel imitation learning settings which requires both tracking and exploration. By moving away from strict temporal path replay and treating demonstrations as informative spatial priors, the proposed method provides a continuous behavioral spectrum that transitions from trajectory tracking to exploration. We leverage the anisotropic diffusion modulated by stagnation to ensure that the target distribution given to ergodic control can induce (i) task geometry-aware, (ii) goal-oriented exploration. When a task-progress mismatch is detected through stagnation metric, the framework leverages geometry-aware anisotropic diffusion to dynamically expand the target particle distribution primarily along directions orthogonal to the reference path. This adaptive target then drives a first-order geometric integrator unrolled directly on the SE(3), ensuring that the planned trajectory respects the underlying nonlinear constraints of the robot's workspace.

Preliminary experiments demonstrated that by autonomously expanding its search landscape around the expert context, the ergodic controller synthesizes novel path corrections to successfully complete the task under spatial perturbations.

5.5.1 Limitations & Future Work

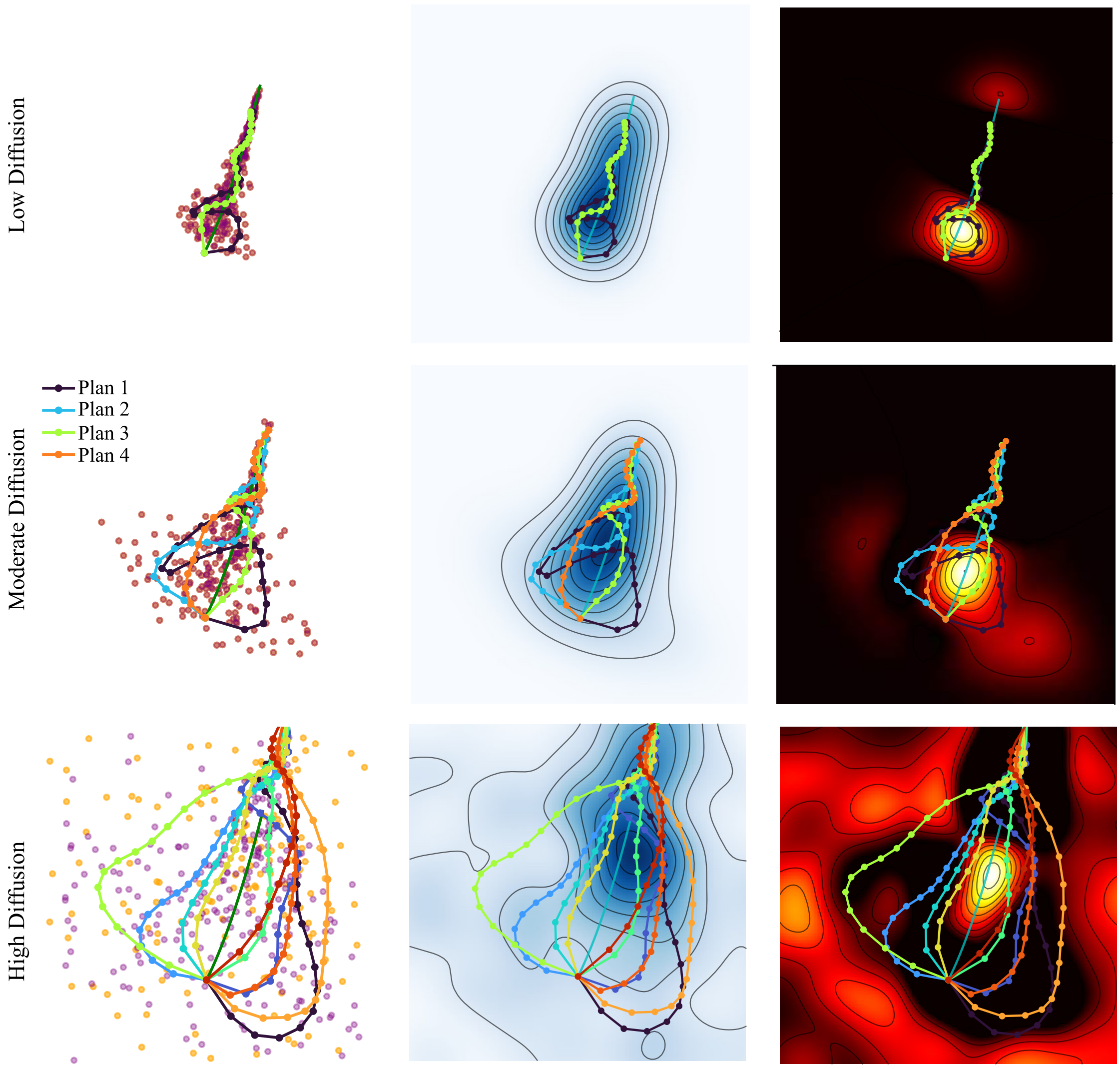
Although we demonstrated the scalability of the framework to SE(3) in simulation, real robot experiments and thorough quantitative evaluations remain crucial next steps. Notably, without prior information regarding which spatial directions to prioritize, unguided exploration in SE(3) can be prohibitively time-consuming. Ideally, the human demonstrator should encode directional variances or target exploration sequences directly into the demonstrations. For example, in peg-in-hole insertion tasks, efficiency could be significantly improved by enforcing a sequential curriculum: first exploring translational degrees of freedom to locate the initial opening, and subsequently shifting to rotational exploration to achieve alignment.

Another limitation of the current framework and kernel-based methods in general is the limited scalability of the coverage memory capacity T_m. This constrains the agent's ability to explore effectively under large distribution shifts. Future work could investigate more efficient memory management strategies, such as using random fourier features or hierarchical memory architectures, to enable more exhaustive exploration.

Additionally, the current framework relies on a simple phase lag estimation module based on nearest-neighbor search from the demonstration dataset. A more robust and generalized alternative would involve a specialized learning module such as [58, 21, 58] that predicts task progress and stagnation based on a richer set of sensory inputs such as proprioception, vision, and tactile feedback.

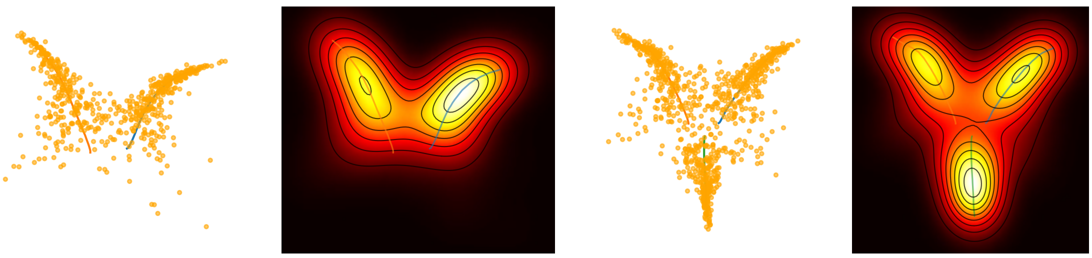
While the proposed framework is agnostic to the underlying action-chunk generation mechanism, for simplicity we demonstrated its efficacy using a retrieval-based imitation learning baseline. A promising immediate direction for future research is to interface this architecture directly with deep generative policies, such as Diffusion Policy or Flow Matching architectures, to evaluate scaling performance on high-dimensional, vision-conditioned tasks.

Finally, Stein Variational Gradient Descent (SVGD) [61, 56] presents a compelling mathematical avenue to unify the decoupled steps of anisotropic diffusion-based target generation and MMD-based ergodic control into a single, integrated optimization problem. By leveraging a matrix-valued Stein kernel structurally scaled by the local geometric tensor, the variational particle updates can be intrinsically warped to enforce attraction and self-repulsion forces directly along task-relevant submanifolds. This mathematical consolidation eliminates the need for a two-stage sampling and tracking pipeline, allowing the receding-horizon control sequence $\mathcal{U}_t$ to be optimized directly via the anisotropic Stein variational gradient in a highly parallelized, JAX-compatible formulation.



(a) Particle Distribution $\mathcal{Q}_t$ (b) Kernelized Target (c) Kernelized Residual

Figure 5.5: Exploration plans with increasing amount of anisotropic diffusion. *Rows:* Increasing diffusion modulator θ. (a) Anisotropically diffused particle distribution, (b) heat kernelized target distribution perceived by ergodic controller using $\tau = 0.01$, (c) coverage residual.



(a) Bimodal Particles (b) Bimodal Kernel (c) Trimodal Particles (d) Trimodal Kernel

Figure 5.6: Anisotropically diffused particle distribution and heat kernel smoothed distribution generated from multimodal expert demonstrations.

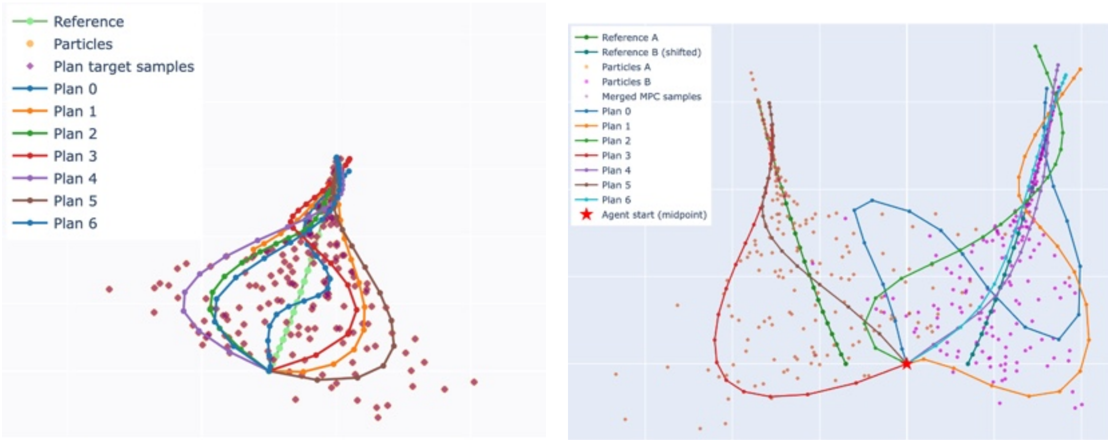
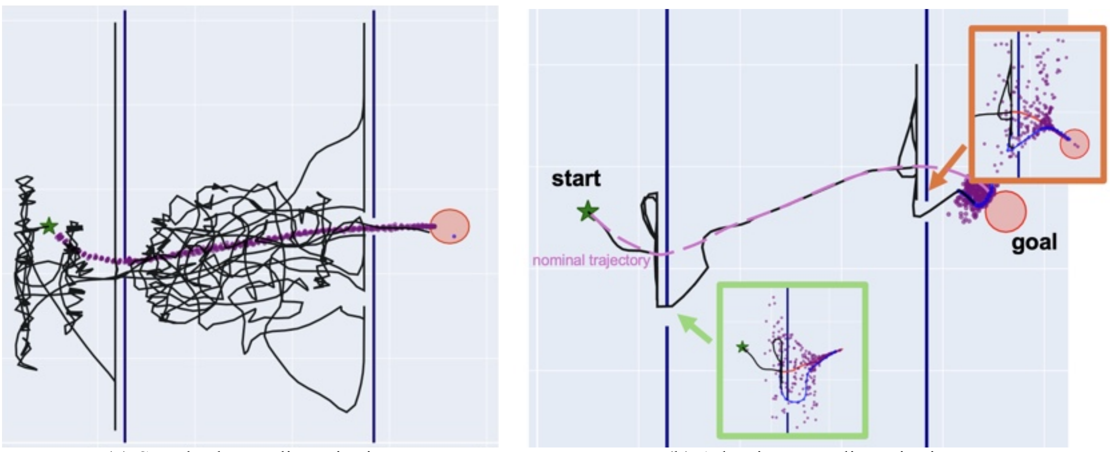


Figure 5.7: Comparison of multiple planning steps using the same target particle distribution with *left:* unimodal reference and *right:* multi-modal reference demonstrations.



(a) Standard Ergodic Imitation

(b) Adaptive Ergodic Imitation

Figure 5.8: Adaptive exploration in the maze environment. (a) Standard ergodic control (SMC) baseline. (b) The proposed adaptive ergodic imitation framework, adaptively switching between tracking and exploration.

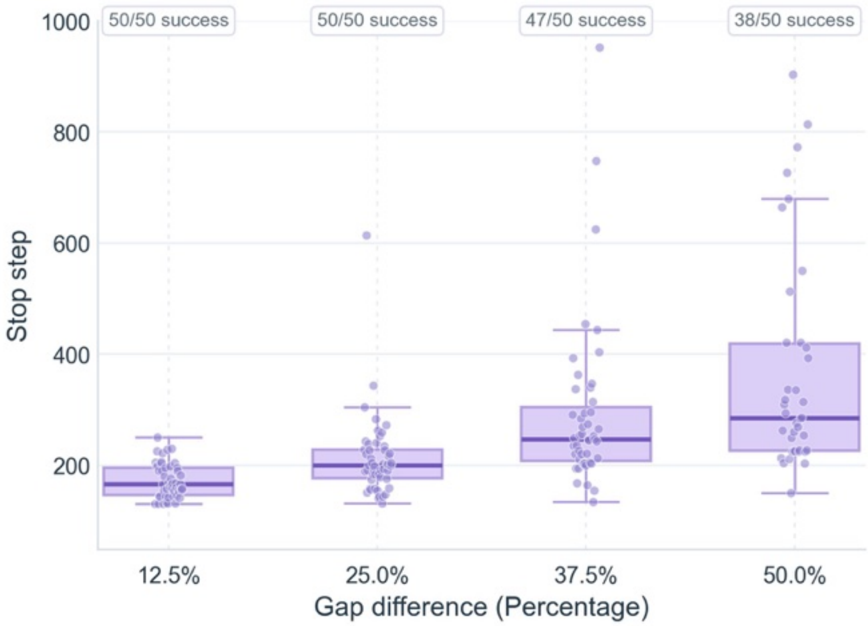


Figure 5.9: Quantitative maze results under gate-location perturbations sampled around the nominal layout.

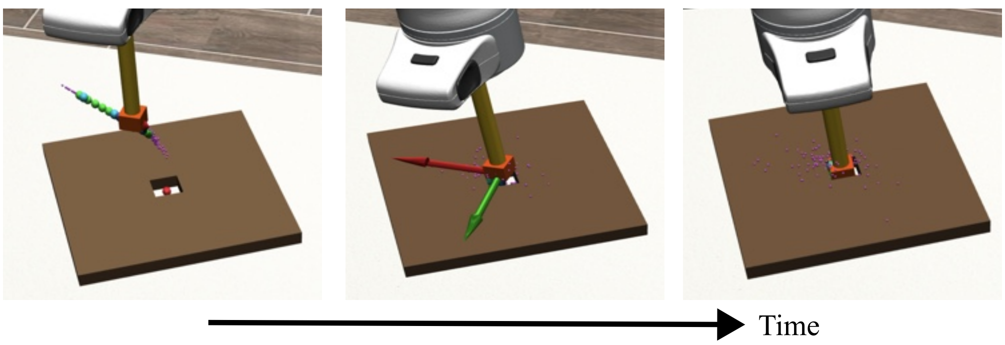
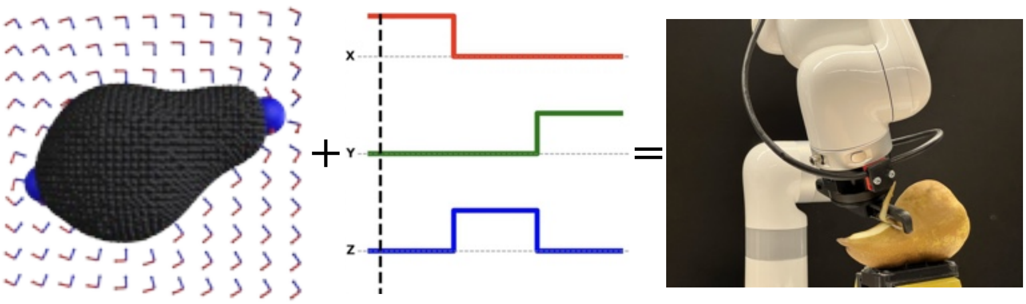


Figure 5.10: Sucessful insertion in SE(3) with position and orientation mismatch.

6 Object-centric Task Representation and Transfer using Diffused Fields



Diffused Orientation Fields | Local Actions | Object-centric task (peeling)

In this chapter, we present a smooth geometric representation of object-centric local reference frames for encoding and transferring manipulation skills across objects. Our representation is computed online from raw point cloud data using diffusion processes governed by partial differential equations, conditioned on keypoints. We evaluate our method under geometric, topological, and keypoint perturbations, and demonstrate our representation in control, trajectory optimization and learning settings.

Publication Note

The material presented in this chapter is adapted from the following publication.

- **Bilaloglu, C.**, Löw, T., and Calinon, S. (2026). Object-centric task representation and transfer using diffused orientation fields. *Science Robotics*

Supplementary Material

Supplementary materials related to this chapter are available at: https://github.com/idiap/diffused_fields_robotics

6.1 Introduction

Humans routinely interact with curved objects as part of daily life, for example, peeling a cucumber, slicing a banana or washing dishes. These tasks require continuous physical interaction with the object's surface, often involving making and breaking contact at different regions. In such tasks, motion on (or near) the surface is guided by the pose and geometry of the object. We refer to these interactions as object-centric tasks. Although, object-centric tasks are challenging for robots due to the unbounded variation in real-world object shapes, humans demonstrate a remarkable ability to transfer learned skills across objects with different geometries. Once a skill is acquired, people can intuitively adapt their motion to new instances, even when the underlying shape changes substantially. We argue that this adaptability arises from using shape-invariant task descriptions paired with shape-aware representations that conform to the specific geometry of the object. This also directly affects what information should be learned and transferred [47]. For example, by learning trajectories using a fixed reference frame, there is the risk of overfitting both pose and object-specific details that can act as noise and hinder generalization. Such pose and shape-dependent details should indeed not be transferred to other objects.

A straightforward approach for pose-invariant skill representation is to define motions in the object's local reference frame [97]. More generally, encoding demonstrations from multiple reference frames enables extraction of task-relevant variability [18]. Using multiple reference frames has been extended to a dictionary of coordinate systems including cylindrical and spherical frames [96], allowing selection of optimal geometric perspectives. Alternatively, coordinate-free descriptors [100, 101, 91] aim to capture invariants of motion without referring to coordinate systems. However, these pose-invariant approaches do not consider the geometry of the object and typically overlook inter-category shape variation.

In contrast, keypoint-based methods [66] describe manipulation tasks using geometric costs and constraints on selected landmarks. These offer robustness to intra-category variation and surface topology changes by filtering out irrelevant geometry. Extensions such as oriented keypoints combine local reference frames with keypoints to support action specification and closed-loop policies [33] and more recent methods leveraged keypoints for visual imitation learning [32, 31]. Neural Descriptor Fields [89, 90] go further by encoding local task descriptors into pose-invariant neural fields, whereas recent work leverages foundation models for learning task descriptors [102]. Instead, policy learning approaches learn perception-to-action mapping from sensory inputs such as RGB [23] or point clouds [109]. Recent reactive-compliance variants explicitly integrate tactile/force feedback to stabilize contact [39, 103]. However, these approaches are fundamentally data-driven and rely heavily on demonstrations and offline training. Ultimately, they aim to learn end-to-end policies rather than a modular, geometry-aware representation that can be reused across objects, controllers, and tasks.

Our key insight is that object-centric tasks like cleaning or slicing on a planar surface can often be reduced to simple, repetitive motions—such as up/down or back/forth—thanks to the presence of a consistent global frame. In contrast, curved or irregular objects lack such a global reference frame, making task specification and transfer more challenging. Oriented keypoints [33] or using multiple body-fixed reference frames [18] can provide geometric structure for local interactions, however they are insufficient for continuous interactions with breaking and making contact near the surface. For that purpose, we construct a smoothly varying continuous representation of local frames throughout the robot's workspace conditioned on surface geometry and a few semantic keypoints. These local frames act as a geometric scaffold for expressing manipulation actions. For instance, a slicing or probing task might be described using a pose- and shape-invariant task description "slide along the object, go down and go up": an instruction that remains well-defined across object instances, categories, and poses. Therefore, our approach to represent and transfer object-centric tasks consists of orientation fields representing a smooth field of local reference frames conditioned on the object geometry and shape-invariant local actions defined in those frames. These local actions can be generated by various high-level controllers ranging from human in teleoperation to trajectory optimizer in planning and a policy programmed using primitives or learned using reinforcement learning.

Our method reduces task transfer across objects to computing orientation fields from online vision and depth input while reusing the same shape-invariant local actions. We demonstrate that our approach provides a shape-invariant representation for tactile tasks such as peeling, inspection, and slicing, enabling transfer to previously unseen objects at runtime. The representation is computed online, robust to input noise, and modular to be integrated with different control, planning, or learning frameworks. Our contributions can be summarized as follows:

- expressing object-centric manipulation tasks as shape-invariant actions in object-centric local reference frames
- representing local reference frames as a smooth orientation field conditioned on point clouds and keypoints collected online
- formulating the smooth orientation field using diffusion processes
- formulating the discretization-free online computation of the orientation field by combining surface and workspace diffusion processes.

6.2 Method

Our method consists of two main components: (i) a representation of local reference frames as a smooth orientation field conditioned on the object geometry and keypoints, and (ii) shape-invariant local actions defined in those local reference frames. In the following sections, we first describe how to compute the orientation field from point clouds and keypoints using diffusion processes. Then, we show how this representation can be used for transferring local actions across objects with different geometries.

6.2.1 Smooth Orientation Fields

In the following subsections, we describe how we combine the diffusion and Laplace's equations with object point clouds and keypoints to generate orientation fields. First, we present computing orientation fields on object surfaces using surface diffusion conditioned on the keypoints. Then, we show propagating these orientations from the object surface to the robot's workspace by solving Laplace's equation (Eq. (2.8)) in the surrounding Euclidean space. Lastly, we describe the computation of the orientation fields in the scenes using multiple surfaces with different representations such as a point cloud, a mesh, or a geometric primitive.

6.2.2 Diffused Orientation Fields on Surfaces

We measured the object surfaces at runtime as point clouds $\mathscr{P}$ composed of $N_{\mathscr{P}}$ points

$$\mathscr{P} := \left\{ (\boldsymbol{x}_i) \;\middle|\; \begin{array}{l} \boldsymbol{x}_i \in \mathbb{R}^3 \\ \text{for } i = 1, \ldots, N_{\mathscr{P}} \end{array} \right\} \tag{6.1}$$

where $\boldsymbol{x}_i$ is the position of the i-th surface point in Euclidean space. In order to solve the diffusion or Laplace's equation on the point cloud we computed the weak Laplacian $\boldsymbol{C}$ and the corresponding mass matrix $\boldsymbol{M}$.

Orientation Fields from Keypoints

The second input of our method is a set of keypoints on the surface $\mathscr{P}_{\text{key}} = \{\boldsymbol{x}_i\}_{i=1}^{N} \subset \mathscr{P}$, which can be either specified by the user, extracted by the perception system or transferred using a vision encoder (see Supplementary Material). We use the keypoints for aligning the orientation field with task-relevant directions. We classify each keypoint either as a "sink" acting as an attractor or a "source" acting as a repulsor. Our strategy for closed surfaces is to use one source and one sink placed as two poles (see Fig. 6.1C, i-ii). This produces a consistent direction field defining longitudes and latitudes for the object and it is easy to extract and transfer using perception modules. If the object has a boundary, we consider all the boundary points as sources (see Fig. 6.1C, iii). In more complex scenarios where we have arbitrary target and/or obstacle regions on the surface, it is possible to set sources and sinks accordingly (see Fig. 6.4B, ii). Next, we use the keypoints to set the initial condition $\boldsymbol{u}_0$, where the values are assigned using an indicator function, setting entries to $+1$ for source keypoints, -1 for sink keypoints, and 0 elsewhere:

$$\boldsymbol{u}_0 = \begin{cases} +1 & \text{if } \boldsymbol{x} \in \mathscr{P}_{\text{source}}, \\ -1 & \text{if } \boldsymbol{x} \in \mathscr{P}_{\text{sink}}, \\ 0 & \text{otherwise.} \end{cases} \tag{6.2}$$

Here we solve the diffusion equation (Eq. (2.10)) to propagate the information from the keypoints across the surface. In the discrete setting, one can solve the diffusion equation by integrating using the implicit time-stepping scheme in Eq. (2.15) where τ is the diffusion time, $\boldsymbol{u}_0$ is the initial condition and $\boldsymbol{u}_\tau$ is the diffused field arrays indexed by the points in $\mathscr{P}$. Diffusion time τ is a hyperparameter that controls the smoothness of the field and is unrelated to wall clock time or to the physical time t used for robot control. In other words, large values for the diffusion time parameter emphasize global structure and symmetries of the object. By denoting $\{\phi_k\}_{k=0}^{\infty}$ the Laplace eigenfunctions on the domain with eigenvalues $\{\lambda_k\}_{k=0}^{\infty}$, the diffusion solution can be described with $u(x,\tau) = \sum_{k=0}^{\infty} e^{-\lambda_k \tau} c_k \phi_k(x)$, where c_k are the coefficients for the initial field $u(x,0)$ in the Laplacian basis. As τ increases, high frequency modes (larger λ_k) are suppressed more quickly by the factor $e^{-\lambda_k \tau}$, so that the result is dominated by low-order modes. On shapes with bilateral or near rotational symmetry, these low-order modes respect the symmetry group, therefore the field aligns with global symmetry as τ grows.

To remove dependence on spatial scale, by following common practice [26], we normalize the parameter by the squared mean neighbor spacing h^2 of the point cloud,

$$\tau \leftarrow \frac{\tau}{h^2}. \tag{6.3}$$

Solving the linear system in Eq. (2.15) is very efficient since it reduces to sparse matrix multiplication if one pre-computes the $\boldsymbol{C}$, $\boldsymbol{M}$ and matrix factorization after obtaining the point cloud (see Table B.3). Next, we compute the gradient of the diffused field $\nabla_{\boldsymbol{x}} \boldsymbol{u}_\tau$ to obtain a smooth direction field on the surface. We visualize these direction fields using red arrows in Figure 6.2B. The direction vector at a given point $\boldsymbol{x}_i$ lies on the tangent plane of the surface at that point $\mathscr{T}_{\boldsymbol{x}_i}\mathscr{P}$. The tangent plane is spanned by two perpendicular vectors $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$. We use the gradient of the diffused field as the first tangent vector $\boldsymbol{e}_1 = \nabla_{\boldsymbol{x}} \boldsymbol{u}_\tau$. Then one can find the perpendicular tangent vector using the local surface normal $\boldsymbol{e}_3 = \hat{\boldsymbol{n}}$ and the cross product $\boldsymbol{e}_2 = \boldsymbol{e}_3 \times \boldsymbol{e}_1$. These three unit vectors are by construction orthogonal to each other and they define both a local reference frame and a proper rotation matrix with positive determinant if they are sequenced in the correct order $\boldsymbol{e}_1$, $\boldsymbol{e}_2$ and $\boldsymbol{e}_3$.

We prefer to use the diffusion equation (Eq. (2.10)) primarily because it allows us to control the smoothness of the orientation field by adjusting the diffusion time τ. This is a critical aspect since using short-time diffusion results in gradient of the diffused field $\boldsymbol{e}_1$ being parallel to the gradient of the geodesic distance (see [26]). In contrast, using long-time diffusion extracts the global symmetry of the objects (see Fig. 6.5A, ii and [60]). Both of these properties can be useful for different tasks and can be adjusted by a single parameter: the diffusion time τ. Secondly, using the diffusion equation let us set the keypoints as initial conditions allowing us to use pre-factorized matrices when solving Eq. (2.15). On the contrary, in Laplace's equation, we need to use boundary conditions altering the discrete Laplacian hence disabling the pre-factorization.

6.2.3 Workspace Diffusion

At this stage, we assume that the orientation field has already been computed on the surfaces using either scalar or orientation diffusion depending on the task. Now we extend these fields from surfaces to arbitrary points in the workspace. Although one could apply the same orientation diffusion procedure by discretizing the workspace and computing its discrete Laplacian, we instead adopt a Monte Carlo method called Walk on Spheres (WoS). WoS avoids the need for discretization, making it better suited for complex, dynamic scenes and generalizable to any surface representation that supports closest point queries.

WoS is a grid-free Monte Carlo method for solving Laplace's equation (Eq. (2.8)) in Euclidean spaces. A function which satisfies Laplace's equation is called a harmonic function and it satisfies the mean-value property, meaning that $u(\boldsymbol{x}) = \mathbb{E}[u(\boldsymbol{X})]$, where $\boldsymbol{X}$ is a random point on a sphere centered at $\boldsymbol{x}$ with radius r. Accordingly, we can simulate a random process where a particle starts at $\boldsymbol{x}$ and jumps to a new position randomly chosen on a sphere centered at $\boldsymbol{x}$. This process repeats until the particle reaches the boundary, defined in our setting as the surface of the object, and the function is assigned the value at the specific boundary point where the particle ultimately lands. We refer the readers to [79] for a thorough analysis on WoS.

The WoS algorithm depends on efficient closest point queries. To enable this, we construct a k-d tree from the point cloud with time complexity $O(n \log n)$ during preprocessing. Each query during runtime then has average complexity $O(\log n)$ and can be efficiently parallelized on the GPU. Unlike the Laplacian-based methods, WoS does not compute the whole diffused field but computes the values of the diffused field in parallel at the query points. In each WoS query, Monte Carlo samples start from the query point in the workspace $\boldsymbol{x}_q \in \Omega$ and converge to boundary points on surfaces $\mathbf{x}_i \in \mathscr{P}$. To compute the value of the diffused orientation field at the query point, we need to average the orientations associated with the boundary points.

For averaging, we represent the retrieved orientations at points $\boldsymbol{x}_i$ as a set of unit quaternions $\mathscr{Q} = \{\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n\}$ and employ the method of Markley et al. [67], which solves the following optimization problem:

$$\overline{\mathbf{q}} = \arg\max_{\mathbf{q} \in \mathbb{S}^3} \mathbf{q}^\top \boldsymbol{S} \mathbf{q}, \tag{6.4}$$

where the matrix $\boldsymbol{S}$ is constructed using the outer product of the unit quaternions:

$$\boldsymbol{S} \triangleq \sum_{\mathbf{q} \in \mathscr{Q}} \mathbf{q}\mathbf{q}^\top. \tag{6.5}$$

Note that Eq. (6.4) is equivalent to maximizing the Rayleigh quotient and its solution is given by the eigenvector of $\boldsymbol{S}$ with the largest eigenvalue

$$\boldsymbol{S}\overline{\mathbf{q}} = \lambda_{\max}\overline{\mathbf{q}}. \tag{6.6}$$

We provide a conceptually simple approximation for computing orientation fields using vector diffusion and orthonormalization in the Supplementary Materials.

In addition to unknown objects collected at runtime as point clouds, complex scenes may include other surfaces represented as meshes or approximated using geometric primitives such as lines, planes, or spheres. By combining the closest point queries for each surface in the scene we can compute the resulting orientation fields in the workspace considering all the surfaces together.

For mesh-based surfaces, we apply the same procedure as for point clouds: we first compute the diffused orientation field on the surface using the discrete Laplacian defined on the mesh, and then construct a boundary volume hierarchy (BVH) for efficient closest point queries. For primitive shapes, we instead use analytical solutions: we compute the surface diffusion using the closed-form Laplacian of the primitive and perform closest point queries using the primitives' analytical distance functions. We provide examples of complex scenes with multiple objects and representations in Figure 6.6.

6.2.4 Shape-Invariant Local Actions

Object-centric local reference frames represented by orientation fields encode task-relevant directions across the workspace considering the surface geometry and semantic keypoints. Expressing tasks in these local frames decouples task representation from the object geometry and yields structured action sequences (Fig. 6.1A). We refer to these shape-invariant task representations that remain unchanged across variation in object geometry as "local action primitives".

Local action primitives are simple closed-loop policies expressed in object-centric local reference frames. At each trajectory point $\boldsymbol{x}_t$, the controller queries the WoS to obtain a local orthonormal coordinate frame $\mathbf{F}_{\boldsymbol{x}} = [\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3] \in SO(3)$ from the diffused orientation field, where $\boldsymbol{e}_1$ is the direction towards the keypoints parallel to the surface (gradient of the smoothened geodesic distance to the keypoints along the surface levelsets), $\boldsymbol{e}_3$ is the direction towards the surface (gradient of the smoothened distance field to the surface) and $\boldsymbol{e}_2$ is the direction orthogonal to $\boldsymbol{e}_1$ and $\boldsymbol{e}_3$ (preserves the smoothened geodesic distance to the keypoints on the surface levelsets and smoothened Euclidean distance to the surface).

The controller computes the next position by stepping according to the action $\boldsymbol{a}_t$ expressed in the local reference frame

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \delta \mathbf{F}_{\boldsymbol{x}} \boldsymbol{a}_t, \tag{6.7}$$

where δ is the step size. Each primitive selects the action based on its motion phase (slide along, go down, lift up, and so on), as shown in Fig. 6.1B. The transition between phases is determined by the distance to the keypoints and by the distance to the surface.

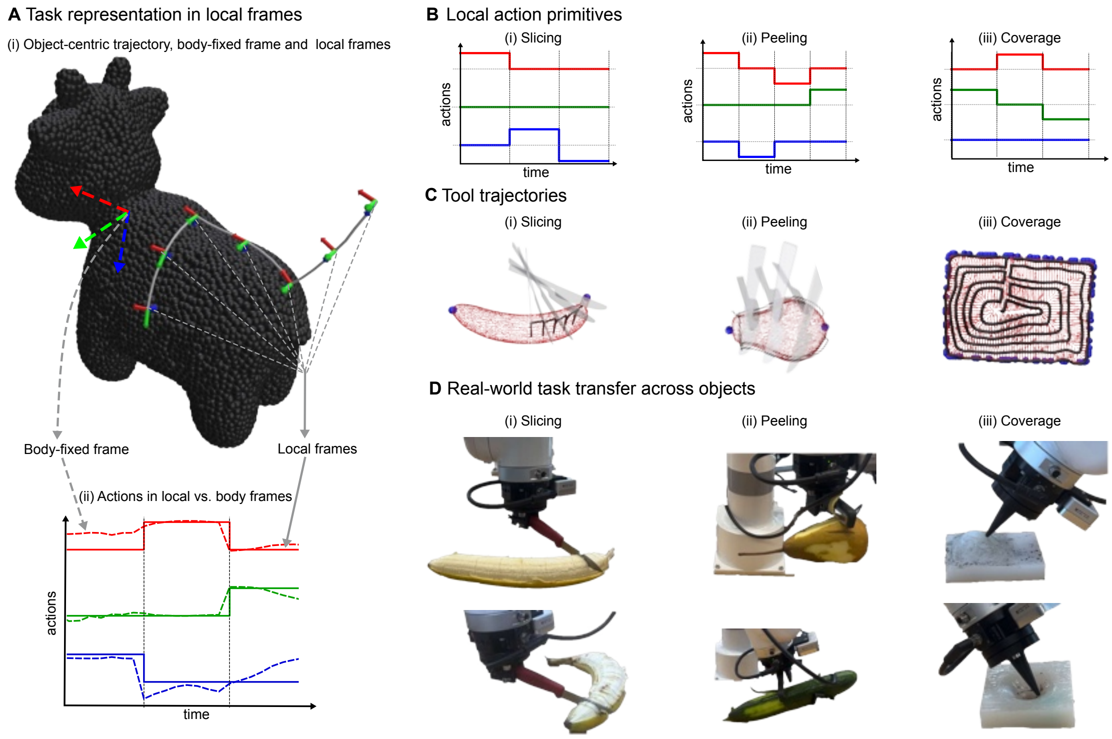


Figure 6.1: Shape-invariant task representation and transfer using orientation fields. (**A**) (i) An object-centric trajectory (gray) composed of an approaching phase and a surface-following phase. Small reference frames are the object-centric local reference frames represented by diffused orientation field, while the large dashed frame denotes a single body-fixed frame. (ii) Gray trajectory is expressed in object-centric local frames (solid lines) and the single body frame (dashed lines). (**B**) Local action primitives providing shape-invariant task descriptions in object-centric local reference frames for (i) slicing, (ii) peeling and (iii) coverage tasks. (**C**) Simulated visualizations showing the keypoints (blue), tool paths (black), and the x-axis of local frames (red) for (i) slicing, (ii) peeling and (iii) coverage tasks. (**D**) Real-world experiments showing transfer of (i) slicing, (ii) peeling, and (iii) coverage tasks to previously unseen objects.

6.3 Experiments

Our experiments aim to address a central question: could our method reduce the complexity of control, planning, and learning on curved objects and enable online transfer?

6.3.1 System Overview

Our system comprised a six-degrees-of-freedom uFactory Lite 6 robot an Intel RealSense D415 stereo camera, a BotaSys SensONE force/torque sensor, and 3D-printed tool mounts for a knife, peeler, and probe (Fig. 6.1D). The inputs to our method were the point cloud collected online and a set of keypoints. Using the object point cloud and the keypoints, our representation provided smoothly varying local reference frames at any point in the workspace. Fig. 6.2A illustrates the workflow, and the full procedure is described in Materials and Methods. Different high level controllers queried the local reference frame at a given position and generated local action references for various downstream tasks. These local action references were then tracked by an admittance controller [80]. The overall workflow is depicted in Fig. 6.2B.

6.3.2 Object-centric Task Representation and Transfer

We transferred tasks represented as local actions to previously unseen objects by computing orientation fields conditioned on the object's point cloud and by tracking local actions using an admittance controller in local frames. We visualized the tool trajectories for a given object and task in Fig. 6.1C and demonstrated real-world transfer across objects in Fig. 6.1D.

To evaluate task transfer quantitatively and compare our approach to alternative object-centric representations, we transferred the peeling task to 50 randomly deformed versions of the Pear object (Fig. 6.3A) from the Yale-CMU-Berkeley (YCB) dataset [19]. We generated deformed object instances by applying random anisotropic scaling, quadratic bending, and twist deformations. For each instance, we first sampled independent scale factors $s_x, s_y, s_z \sim \mathscr{U}(0.6, 1.4)$ and scaled the coordinates along the corresponding axes. To introduce nonuniform deformations, we then applied quadratic bending along each Cartesian axis. For bending about axis $j \in \{x, y, z\}$, we sampled a curvature parameter $c_j \sim \mathscr{U}(-3, 3)$ and displaced the coordinates along the two orthogonal axes $i \neq j$ according to

$$p_i \leftarrow p_i + c_j\, p_j^2. \tag{6.8}$$

In addition, we applied twist deformations around each axis. For twisting about axis j, we sampled a twist strength $\alpha_j \sim \mathscr{U}(-3, 3)$ and rotated each point around axis j by an angle

$$\theta_j = \alpha_j d_j, \tag{6.9}$$

where d_j is the coordinate of the point along the twist axis.

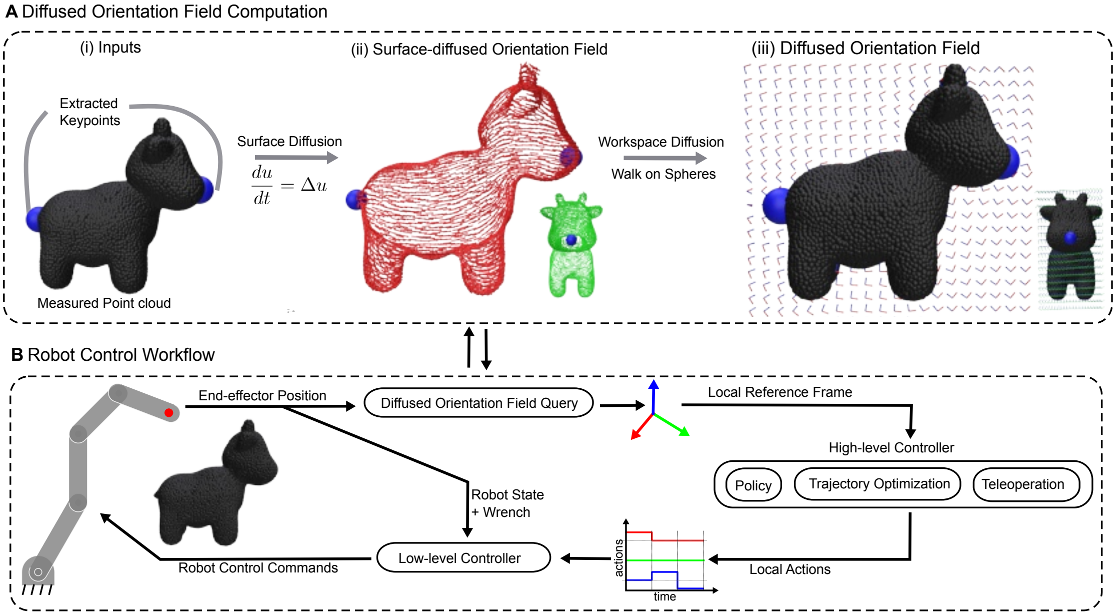


Figure 6.2: Overview of the proposed representation with complete workflow. We use Spot [25] as our canonical point cloud throughout the paper as it provides a sufficiently complex object with a well-defined symmetry axis for cross-section views. (**A**) Overview of the diffused orientation field computation. (i) Our method takes a point cloud collected at runtime and keypoints as its input. (ii) We compute a surface orientation field conditioned on the keypoints by solving the diffusion PDE on the point cloud. We visualize the orientation field by using local reference frames. We show the x-axis in red, the y-axis in green and we omit the z-axis for clarity. (iii) We extend the surface orientation field to any point in the workspace using workspace diffusion. Diffused orientation field represents smoothly varying local reference frames across the workspace by considering the object's surface geometry. We visualize it as a grid to show its smoothness, but in practice, we evaluate the field only at the query position. (**B**) Overview of our robot control workflow using local reference frames and actions. High-level controllers query the orientation field at the robot position to obtain the local reference frame and produce local actions expressed in that frame for downstream tasks. These local actions are provided as references for the low-level tracking controller.

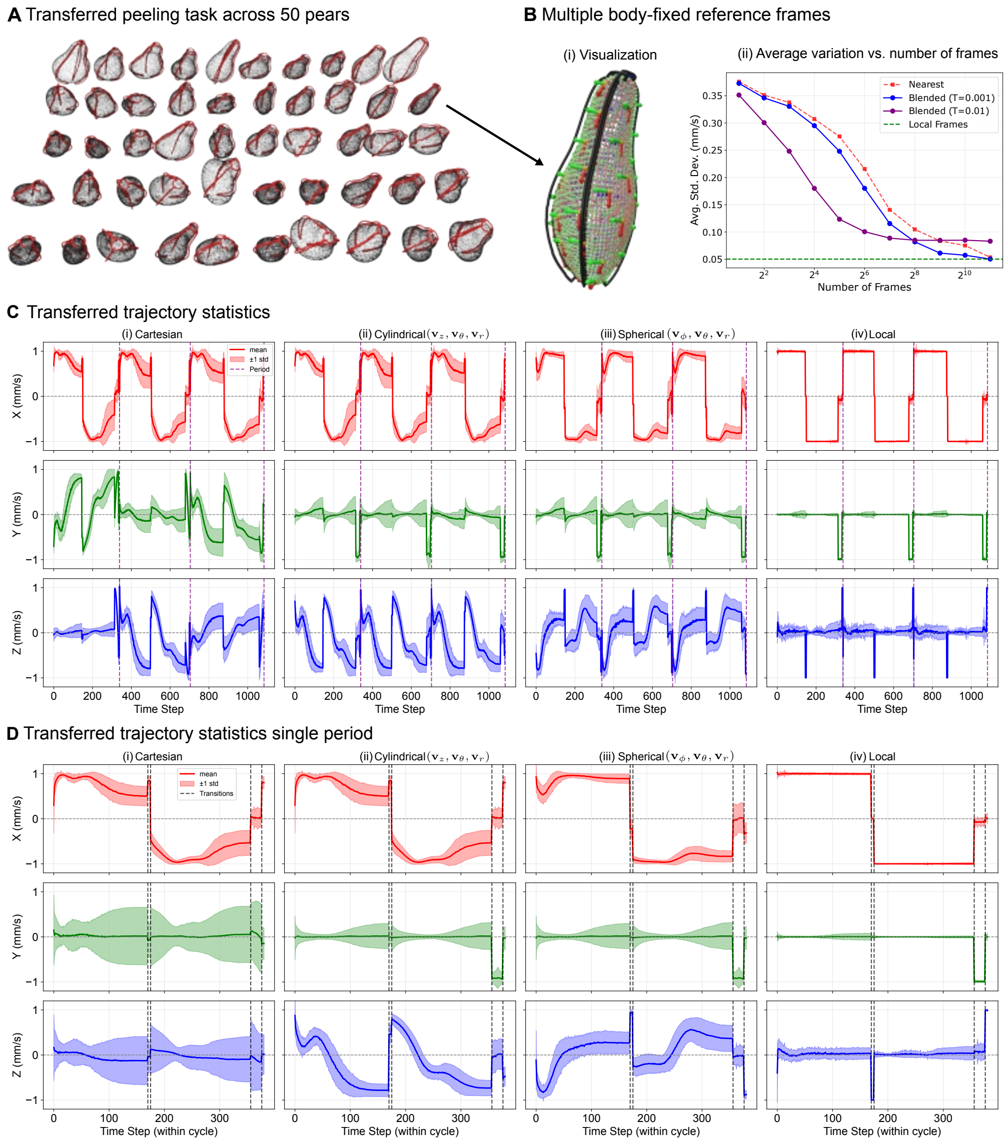


Figure 6.3: Task transfer across objects. (**A**) Transferred peeling trajectories (red) across 50 pear instances (black). Each trajectory consists of three peeling cycles. (**B**) Comparison of using multiple discrete body-fixed reference frames versus a field of local reference frames provided by orientation fields. (i) Visualization of 50 body-fixed reference frames sampled on a pear instance. (ii) Average standard deviation of the transferred action (velocity) trajectories with respect to number of sampled body-fixed reference frames. (**C** and **D**) Comparisons of orientation fields with respect to baselines in terms of the full and period-aligned transferred trajectory statistics, respectively. Action trajectory statistics expressed in (i) Cartesian, (ii) cylindrical, (iii) spherical body-fixed reference frames and (iv) local reference frames provided by orientation fields.

For comparisons, we selected four object-centric representations as baselines: a body-fixed Cartesian frame, a body-fixed cylindrical frame, a body-fixed spherical frame, and multiple body-fixed frames distributed at equal spacing over the object surface (Fig. 6.3B). A single body-fixed frame is the most common object-centric choice because it provides pose-invariance, but it ignores the geometry of the shape. The cylindrical and spherical variants retain pose invariance and additionally encode axial and radial symmetry, respectively [96]. To ensure a fair comparison and take advantage of object symmetry, we aligned each approach with the primary symmetry axis of the object, defined by the line connecting the two keypoints used in our method. We then expressed the actions as linear velocities along the three orthogonal directions induced by each approach.

The multiple body-fixed frames baseline serves two purposes. First, it is a discrete approximation of our approach, since orientation fields yields a continuous field of local frames over the surface. Second, it represents a common family of methods in the literature, including task-parameterized models [18], oriented keypoints [33], and neural descriptor fields [89]. Importantly, our instantiation is an oracle version of these approaches, because it uses exact keypoint correspondences and the task aligned local frames provided by our method. This choice favors the baseline and thus makes the comparison conservative.

As the comparison metric, we used statistics of the transferred action trajectories. An ideal object-centric task representation for transfer should be shape-invariant and therefore produce lower variation across shapes. We reported the mean and the standard deviation of 50 trajectories that each contained three peeling cycles (Fig. 6.3C). Our representation yielded the smallest variation because it decoupled object geometry from the task coordinates. It also preserved the three cycle periodic pattern in all coordinates, which showed that the global symmetry of the object was captured while remaining separate from the task representation. In contrast, the other baselines captured only the particular symmetry they encode and only along one or two directions. To make this effect explicit, we further decompose each trajectory into individual peeling cycles, align them in time, and plot the mean and the standard deviation of the actions for each baseline in Fig. 6.3D.

To show that our method is the continuous version of approaches based on multiple body-fixed reference frames, oriented keypoints, and neural descriptor fields, we measured the average variance of transferred trajectories as a function of the number of frames sampled using Farthest Point Sampling on the point cloud (Fig. 6.3B ii). At each robot position, we expressed velocities in sampled frames using two variants: nearest sampled frame assignment, and distance-weighted blending with softmax weights

$$w_k = \frac{\exp(-d_k/T)}{\sum_j \exp\left(-d_j/T\right)}, \tag{6.10}$$

where d_k is the distance from the robot position to frame k and T is a temperature parameter. As expected, the variation decreased as the number of frames increased, converging to our continuous local frame method.

From an imitation learning perspective, the transferred trajectories can be treated as demonstrations of the peeling task across different object instances. Our representation yielded lower variation, directional decoupling, and periodic structure, which should enable better learning performance for statistical learning methods.

We provide task-level quantitative evaluations against state-of-the-art imitation learning and planning methods in the Supplementary Materials, demonstrating that our representation matched baseline performance on simple shapes and successfully generalized to complex, non-symmetric, and irregular geometries.

6.3.3 Integration with Different Control Paradigms

Diffused orientation fields serve as an intermediate geometric representation that is agnostic to how high-level commands are generated, enabling seamless integration with diverse control paradigms (Fig. 6.4A). In addition to the local action primitives presented in the previous section, we integrated orientation fields in three settings to show it reduces controller complexity by decoupling the action space from object geometry: a teleoperation controller, a planner based on trajectory optimization, and a policy learned with reinforcement learning.

In teleoperation experiments, we used orientation fields to assist the operator by automatically aligning the tool based on the object's surface and task-specific keypoints. We employed the 3DConnexion Space Mouse, a six-degree-of-freedom input device, and mapped its control axes to the object-centric local reference frames. Motion along the input device's x-axis translated to movements that approached or retreated from keypoints while maintaining a constant distance from the surface. Motion along the z-axis controlled approach or retreat relative to the object, and motion along the y-axis preserved the distance to both the keypoints and the surface. Throughout the task, the desired tool orientation, relative to the surface and task direction, was maintained. This setup enabled intuitive, surface-aware teleoperation, as illustrated in Fig. 6.4B.

In order to show how our representation can be more broadly applied in complex especially long-horizon robotic manipulation tasks, we integrated it in trajectory optimization for planning. We used orientation fields to define cost functions and their gradients based on the distance to the object's surface and geodesic distances to surface regions encoded using keypoints. These components enabled gradient-based optimization to compute robot trajectories that both maintained a desired distance from the surface and reached target regions while avoiding obstacles (Fig. 6.4C, i-ii). For simplicity, we adopted a batch formulation of the iterative linear quadratic regulator (iLQR), modeling the robot as a velocity-controlled point mass. To further improve convergence, we warm-started the optimization using geodesic shooting, initializing the trajectory by following the x-axes of the local reference frames (Fig. 6.4C, iii-iv). We evaluated this setup across fifty different initial positions and plotted the number of iterations required for convergence (Fig. 6.4C, v). Results showed that with warm-starting using orientation fields, convergence was typically achieved in a single iteration, whereas

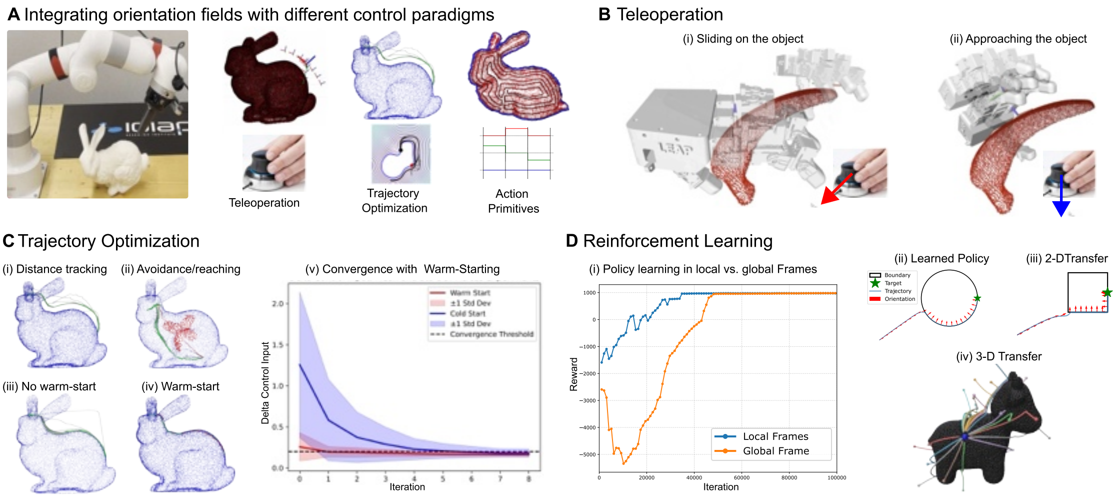


Figure 6.4: Transfer across controllers using orientation fields. (**A**) Orientation fields as a controller agnostic intermediate representation which can be integrated with various reactive and anticipative controllers. (**B**) Teleoperation using a space mouse and the LEAP hand [86], where the input axes are mapped to local frames. (i) Moving along x-axis (red arrow) slides the tool along the surface. (ii) Moving along z-axis (blue arrow) approaches the surface. (**C**) Trajectory optimization experiments for (i) distance tracking, (ii) target reaching and obstacle avoidance, (iii) reaching without warm-starting and (iv) reaching with warm-starting using orientation fields. (v) Norm of the change in the control commands, which is used as the convergence criteria for the trajectory optimization, showing the effect of warm-starting. (**D**) Learning and transferring policies using reinforcement learning in local reference frames. (i) Reward evolution while learning a reaching policy using local and global reference frames. (ii) Learned target reaching and distance tracking policy in local reference frames of a 2-D circle. Zero-shot transfer of the learned policy on the circle to (iii) a 2-D rectangle and (iv) a 3-D point cloud.

without warm-starting, at least five to six iterations were required. Qualitative observations of the initial and final trajectories further supported this finding. In essence, our representation provided a near-optimal solution for distance tracking, reaching, and surface-based obstacle avoidance purely from geometric information.

We ran proof of concept experiments to integrate our representation with reinforcement learning. First, we trained a two dimensional reaching policy and compared using local reference frames versus a global frame, measuring reward evolution over training (Fig. 6.4D, i). Even in this simple setting, providing a geometric scaffold through local frames improves learning performance. Next, we considered a two dimensional reaching and distance tracking task on a circle (Fig. 6.4D, ii). Using the same environment and reward, we trained two policies: a local policy that issues actions in local frames and a body-fixed policy that acts in a single frame attached to the object. We then deployed both policies on previously unseen shapes. The local policy learned in local frames transferred directly to new two dimensional shapes such as a square and to three dimensional objects such as the Spot point cloud (Fig. 6.4D, iii-iv). In contrast, the policy trained in a body-fixed frame did not transfer to other shapes. Additional details on local policy learning are provided in the Supplementary Materials.

6.3.4 Robustness to Sensing Imperfections

In robotic applications, sensor data is often noisy or incomplete, and control or tool positioning may be imprecise. Our representation provides robustness in such scenarios through its inherent smoothness, governed by the diffusion time parameter τ. This parameter controls the extent of smoothing in the orientation field: lower values emphasize local geometric details, whereas higher values produce smoother fields that reflect the global symmetry structure of the shape. It is important to note that diffusion time is only a parameter that controls the smoothness of the field, and is not related to the time required to compute the field. We illustrated the influence of diffusion time on field smoothness in Fig. 6.5A, and compared our method to a baseline using projection to the surface orientation field in Fig. 6.5B.

Although real-world experiments already demonstrate the effectiveness of our method under realistic sensor noise, we further evaluated its robustness under controlled synthetic perturbations. Specifically, we investigated how increasing the diffusion time affected robustness in the presence of various noise types. All experiments were conducted on the Banana point cloud from the YCB dataset [19].

We performed three experiments, each repeated 50 times with randomly sampled noise. For topological noise, we simulated occlusions and limited viewpoints by removing half of the point cloud and introducing ten randomly placed holes with a 5 mm radius, disrupting surface connectivity. For geometric noise, Gaussian noise with standard deviation $\sigma = 3$ mm was added to the 3D point positions. Finally, for keypoint noise, Gaussian noise with standard deviation $\sigma = 20$ mm was applied to both keypoints, which were then projected back onto the nearest point cloud vertices. To quantify robustness, we computed the root mean square

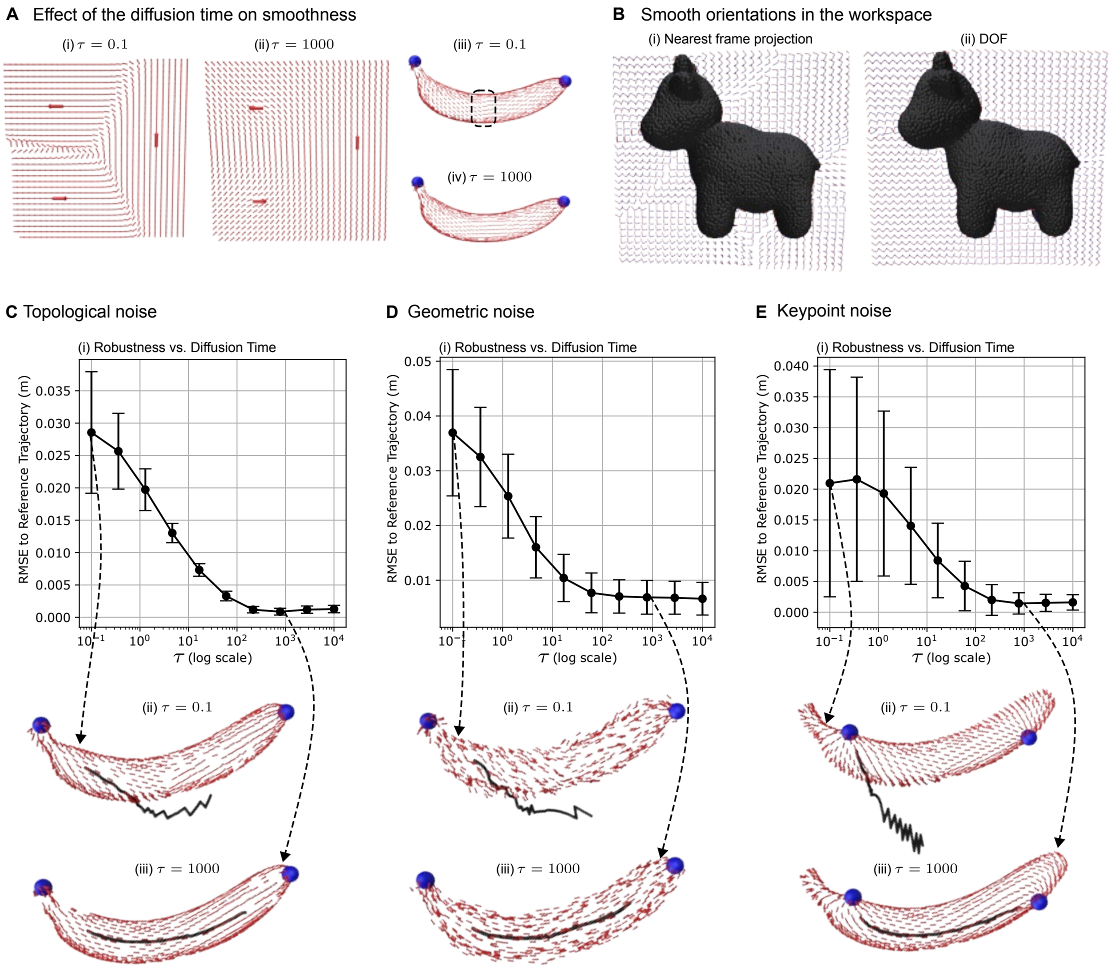


Figure 6.5: Effect of increasing the diffusion time on robustness to noise. We visualize the local reference frames using their x-axes shown in red. (**A**) Effect of the diffusion time on the smoothness of the orientation field. Planar orientation field using keyorientations (large arrows) with (i) short-time and (ii) long-time diffusion. Non-planar orientation field using keypoints (blue points) with (iii) short-time and (iv) long-time diffusion. (**B**) Comparison of projection to surface orientation field versus the smooth orientation field produced by the diffusion. (**C**, **D** and **E**) (i) Error bar plots showing that robustness to topological, geometric and keypoint noise increases with the increased diffusion time. Data represent the mean root mean square error (RMSE) compared to the reference trajectory across $n = 50$ independent trials with randomly sampled noise perturbations on the point cloud. Error bars indicate the standard deviation. (ii and iii) Two instances with noisy inputs from the experiments visualizing the effect of the short and long diffusion times. Red arrows are the x-axes of the local reference frames, the black curve is the trajectory compared to the reference trajectory and the blue points are the keypoints.

error (RMSE) between each resulting trajectory and a reference trajectory. The reference was generated using noise-free inputs, diffusion time $\tau = 1000$, and tracking the local x-direction of the local frames.

As expected from the diffusion equation which suppresses high frequency noise, and as confirmed empirically in our real-world experiments, increasing diffusion time consistently produced smoother and more robust orientation fields across all tested perturbations (Fig. 6.5C–E). However, when depth returns are severely degraded or absent, smoothing alone cannot recover the missing geometry. Specifically, standard depth sensors often struggle with transparent, translucent, or highly reflective objects, resulting in noisy, distorted, or incomplete point clouds. Because our method relies directly on this point cloud data, its real-world execution is inherently tied to the reliability and accuracy of the depth-sensing modality. Consequently, highly reflective or transparent/translucent objects present a practical limitation for our current pipeline. Furthermore, the overall precision of our representation is bounded by the general resolution and noise characteristics of the camera used. Although these represent practical hardware constraints rather than fundamental algorithmic limitations of our shape-invariant approach, they remain important considerations for real-world deployment. In these scenarios, our method is required to use a partial reconstruction/completion method or additional sensors providing better depth data. Additional experiments in clutter, occlusions and with transparent objects are provided in the Supplementary Materials.

6.3.5 Multi-Object Scenes and Compositionality

Our representation is agnostic to the surface representation and the number of objects in the scene. This enables its application in cluttered environments containing multiple surfaces with diverse representations (Fig. 6.6A-B). Interestingly, cluttered workspaces can improve the computational efficiency of our representation due to the specific method used for its computation; such environments tend to result in mostly enclosed regions rather than unbounded ones. For instance, in the setup shown in Fig. 6.6B, introducing an enclosing sphere reduces the computational cost by $\approx 1.5\times$. We discuss this further in the Materials and Methods section.

Diffused orientation fields supports any surface representation, provided that closest-point queries can be computed. In our experiments, we focused on primitive shapes defined by analytical distance functions. In simple cases, primitives can approximate basic surfaces, such as walls, planes, or workspace boundaries, whereas combinations of these shapes, such as spheres and capsules, can compose more complex environments for obstacle avoidance. More interestingly, primitives can encode task constraints directly into our representation. For example, enclosing an object point cloud with a sphere can regularize the orientation field far from the object (Fig. 6.6C).

We modeled a proof-of-concept long-horizon scooping task with multiple objects using our representation (Fig. 6.6D). Task constraints such as keeping the tool horizontal to prevent

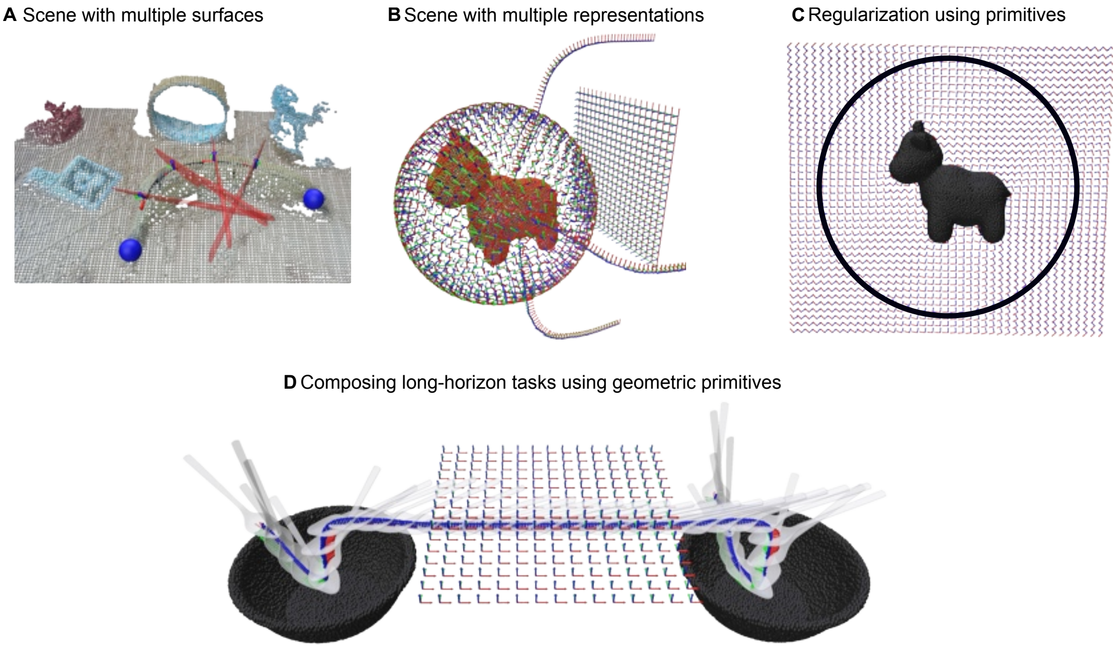


Figure 6.6: Diffused orientation fields constructed from point clouds and primitive shapes. (**A**) Slicing on a banana in a cluttered scene with multiple objects. (**B**) Object point cloud surrounded with an enclosing sphere and a plane designating a wall in the workspace. Trajectories starting from different surface points and following local z-directions to visualize smooth variation of local reference frames across the workspace. (**C**) Cross section of the diffused orientation field around the object, regularized with an enclosing sphere. (**D**) A scoop-lift-carry-pour task using two bowls from the YCB dataset [19], together with lines and a plane to impose task-specific constraints.

spillage and specifying the lifting direction were encoded directly as simple geometric primitives: a plane to enforce level orientation and a line to define upward/downward motion. This construction introduces no extra parameters for combining and requires no behavior tuning. Since diffused orientation fields provide the smoothest orientation field consistent with these constraints across the workspace, the controller is naturally robust to different initial conditions and moderate perturbations.

6.3.6 Comparisons

We qualitatively compared our method against four commonly used representations in robotics. **Primitives** represent objects using simple geometric shapes with analytical distance functions; although computationally efficient and well-suited for collision avoidance, they lack the geometric detail required for fine surface interaction. Alternatively, **Surface Parameterizations** map points on a surface to coordinates in $\mathbb{R}^2$, but they are restricted to the surface itself and inherently introduce distortion due to curvature. We also evaluated **Signed Distance Fields (SDFs)** [29], which implicitly encode the distance to the nearest surface point. Although useful in 3D space and supportive of level set representations, SDFs lack geodesic information. Lastly, **Neural Descriptor Fields (NDFs)** [89] utilize neural networks to learn smooth implicit surface representations from data, yet despite their expressiveness, existing approaches are typically limited to scalar fields and require offline training.

We compared these representations across key properties relevant to object-centric tasks in Table 6.1. The primary advantage of our representation over alternatives is its ability to encode both geodesic and Euclidean information, enabling geometry-aware interactions on and across level sets.

	Primitives	**SDFs**	**NDFs**	**Surface Param.**	**DiffOF**
Geodesic-aware	✓	×	×	✓	✓
Extends to workspace	✓	✓	✓	×	✓
No training required	✓	✓	×	✓	✓
Online updates	✓	✓	×	×	✓
Exact distances	✓	✓	×	✓	×
Tactile tasks	×	✓	✓	✓	✓

Table 6.1: textbfComparison of object representations across key properties relevant to manipulation and planning. We abbreviated diffused orientation fields as DiffOF.

6.4 Conclusion

The diffused orientation fields smoothly represents object-centric local reference frames across the workspace, conditioned on object point clouds and keypoints collected online. Since orientation fields encode surface's geometric structure, it enables shape-invariant task representations and their transfer across curved objects. We demonstrated the transfer capabilities of our representation in real-world experiments with three distinct tasks across six different objects (Fig. 6.1). We assessed the robustness of orientation fields under noise in keypoint extraction, point cloud data (Fig. 6.5). These experiments highlight the inherent robustness of our method, owing to the smoothing effect of the diffusion PDE, making it well-suited for robotics scenarios characterized by noisy and incomplete sensory data. Our results show that our method scales out-of-the-box to complex scenes including multiple objects and diverse surface representations (Fig. 6.6) and its computational efficiency (Table B.3) enables online updates to the field in changing scenes.

We compared orientation fields qualitatively and quantitatively to commonly used object-centric representations in robotics (Fig. 6.3 and Table 6.1). Our representation seamlessly combines the gradient of the signed distance field (SDF) to the surface with the gradients of geodesic distances to keypoints, yielding a single smooth field. This integration allows us to reason about directions pointing to keypoints on the level sets and directions pointing towards the surface. In contrast to orientation fields, learning-based approaches such as neural descriptor fields rely on data-driven learning to implicitly capture structures, offering greater expressiveness but at the cost of requiring training data. Our approach takes a different stance: it treats keypoints as encoding the inductive biases relevant to the task and propagates this structure across the workspace using a diffusion process. This reframes task transfer across objects as the simpler problem of transferring keypoints. This trade-off is often advantageous since keypoints can be extracted using simple perception pipelines such as boundary detection, transferred through foundational models as shown in prior work, or manually annotated when needed, given that they form a sparse and interpretable set.

Given that orientation fields is designed for manipulation tasks that can include dynamic objects, we evaluated its suitability for online computation (Table B.3). We observed that the most expensive operation is the construction of the Laplacian operator from the object's point cloud, computed during preprocessing. At runtime, this operator can be reused even under isometric deformations encapsulating rigid body transformations and bending without stretching. Although we do not evaluate this explicitly in the current paper, this property makes orientation fields applicable to deformable objects such as garments or cables. However, we note that interactions with soft objects alter the underlying geodesic structure, and therefore the precomputed Laplacian may need to be updated to reflect substantial deformations.

6.4.1 Limitations & Future Work

A core limitation of using the Walk on Spheres (WoS) method is its restriction to solving Laplace's equation, which yields the smoothest possible vector field given the boundary conditions. Although this is beneficial for robustness, it also limits expressiveness in scenarios where more localized or structured behavior is desired. One promising direction to address this limitation, without losing the computational advantages of WoS, is to solve the screened Poisson equation instead of Laplace equation [79]. The screening term introduces a tunable decay, enabling better control over the spatial extent and locality of the resulting field.

Another practical limitation of our approach is the need to predefine the tool center point (TCP) before the execution. This requires precise calibration of the tool. In real-world scenarios, especially in tactile tasks, the actual contact point may shift dynamically along the tool during interaction. We showed in experiments that, due to smoothness, diffused orientation fields provides robustness against positioning errors of the robot. Nevertheless, a more robust alternative would involve estimating the effective TCP online by combining vision and force/-torque feedback. Orientation field queries could then be performed at these dynamically inferred contact points, improving reliability during complex manipulation.

Complex object-centric tasks can be composed from local actions expressed in local frames through planning or learning. For planning, trajectory optimization can exploit gradients provided by for costs and constraints that follow the surface and keypoints (Fig. 5C). This can enable planning sequences that alternate between approach, contact, sliding, and retreat while remaining consistent with the object geometry. For policy learning, expressing observations, actions, rewards, and demonstrations in local frames provides pose and shape-invariance (Figs. 4 and 6D). This in turn can allow demonstrations from different objects to be combined into a unified policy, since their local descriptions become comparable under the local frames, and supports transferring a policy learned on one object to another. Moreover, our entire pipeline is differentiable with respect to both the task keypoints and the diffusion time parameter [83, 70]. Therefore a promising future extension is to integrate diffused orientation fields into learning and optimization based frameworks where these parameters are optimized jointly with task performance.

In conclusion, our research demonstrated that task transfer across complex curved objects can be achieved by reusing shape-invariant actions within a smoothly varying representation of local reference frames. By conditioning diffusion processes on the target object's point cloud and keypoints, we developed a computationally efficient and robust method for computing these local frames online, even in the presence of partial and noisy sensor data. Because this approach decouples the action space from specific object geometry, it inherently yields pose and shape-invariance. Ultimately, existing robotic learning and planning pipelines can harness this invariance to considerably enhance their generalization capabilities. As indicated by our proof-of-concept experiments, a promising direction is to utilize our representation to combine demonstrations from entirely distinct object shapes into unified, generalizable policies. Similarly, our work highlighted the potential to train policies on simple simulated geometries and transfer those skills zero-shot to more complex objects.

7 Conclusion

This thesis demonstrates that diffusion processes bridge spatial smoothness with intrinsic geometry, providing an inductive bias for representations that enable online adaptation in robotic manipulation. We validate this framework across tactile exploration and skill-transfer tasks, adjusting to variations in object shape and environmental conditions by leveraging diverse onboard sensing modalities including depth cameras, force-torque feedback, and the robot's entire kinematic chain operating as a whole-body tactile sensor.

A central contribution of this work is demonstrating that while raw sensory feedback is noisy, partial, and degraded by occlusions, leveraging the interdependent structure of geometry and smoothness provides the required regularization. Ultimately, this inductive bias allows the framework to operate on *point-based representations* of non-Euclidean domains. It can account for the multi-link geometry of the robot, accommodate shape variations in everyday objects without requiring complete reconstructions, and respect the underlying geometry of rigid-body motions.

Although diffusion processes are an extensively studied topic across diverse analytical fields, their application as smooth, deterministic fields guiding robotic manipulation has been underexplored. The methodologies developed across the chapters of this thesis demonstrated the versatility of diffused fields in diverse manipulation problems:

- **Coordinating whole-body exploration** by utilizing the manipulator's entire link geometry as a distributed, kinematically constrained tactile sensor;
- **Tactile exploration and mapping on curved surfaces** by solving the governing diffusion PDEs directly on vision-acquired point clouds at runtime;
- **Online adaptation in imitation learning** that builds a continuous behavioral spectrum between rigid tracking and goal-oriented spatial exploration.
- **A shape-invariant geometric basis** that successfully decouples task descriptions from structural variations for object-centric skill transfer;

The methodologies presented in this work demonstrate that autonomous manipulators can tolerate real-world variability, unmodeled dynamics, and sensory uncertainty by relying on structural geometric invariants rather than excessive data aggregation. While these implementations maintain the high computational efficiency required for real-time control loops, they rely on various assumptions to preserve algorithmic tractability. These assumptions define the practical limitations of the proposed framework while systematically charting the trajectory for future research directions.

7.1 Limitations

7.1.1 Loss of Detail due to Excessive Spatial Smoothing

Although utilizing spatial smoothness as a regularizer for noisy and partial data provides an effective inductive bias in most real-world scenarios, certain tasks can be hindered by over-smoothing. For instance, as demonstrated in Chapter 3, employing stationary diffusion can over-smooth the target, resulting in a loss of fine-grained details. This is a recognized trade-off within spectral methods; the diffusion PDE and its fundamental solution (the heat kernel) possess a free parameter, the *diffusion time*, which dictates the extent of spatial smoothing by acting as a coefficient in a multiscale low-pass filter.

To mitigate this limitation, this smoothing parameter could be learned from data [83]. Alternatively, one could replace the standard heat kernel with mathematical variants that introduce additional parameters to modulate smoothness, such as generalized Matérn kernels [16].

7.1.2 Simplified Robot Model

The theoretical formulations throughout this thesis primarily treat the robot and target objects as kinematic entities, assuming negligible system dynamics and a singularity-free task workspace. Furthermore, physical interactions are largely modeled through localized abstractions:

- **Kinematic Robot Control:** The high-level planners and operational-space controllers developed in this work generate impedance or admittance reference targets in the task space. This paradigm presumes that the low-level controllers can compensate for robot dynamics, friction, and inertial effects.

- **Tool Center Point (TCP) Abstraction** Except for the whole-body control method introduced in Chapter 3, we rely on a rigidly calibrated, single-point TCP. In realistic tactile manipulation tasks such as cleaning, peeling, or slicing, physical contact is inherently multi-point and continuously migrates across the tool's surface geometry. That being said, because our method leverages smoothly varying fields, a minor deviation in the calibrated Tool-Center-Point (TCP) results in a correspondingly small variation in the field value, thereby affording the system a baseline degree of geometric robustness. However, in scenarios where these alignment errors exceed this inherent tolerance, we recommend integrating force/torque or tactile feedback with tool meshes or real-time vision models to dynamically update the query coordinates evaluated on the diffused field.

- **Kinematic Singularities** Operating directly via task-space diffused fields assumes that all spatial coordinates are kinematically reachable. However, fixed-base manipulators are strictly bounded by configuration-dependent singularities and joint limits. While geometric limits can be encoded directly into the diffusion PDE using Neumann boundary conditions to repel the robot from workspace boundaries, joint-space singularities require coupling our task-space representations with reactive joint-limit avoidance frameworks, such as geometric fabrics [98, 92] or null-space projection techniques.

7.1.3 Static, Single-View Point Cloud Assumptions

A practical limitation of our current experimental validation is the assumption of a static environment where the target object remains fixed, and its point cloud is captured from a single initial perspective. In more realistic, unconstrained scenarios, a robot might perform bimanual manipulation, holding and reorienting an object with one hand while operating on it with the other.

Scaling the diffusion processes to these settings introduces two critical technical requirements: (i) verifying whether the manipulated object across sequential frames is identical up to an isometric transformation or if a novel object has been introduced, and (ii) if it is recognized as the same object, dynamically aggregating and registering multi-view point clouds online to maintain a coherent spatial representation. Although a detailed evaluation of this closed-loop dynamic perception falls outside the scope of this thesis, we provide preliminary proof-of-concept integrations leveraging vision-language models like Grounded-SAM [77] and robust visual descriptors such as DINO [75] for automated semantic segmentation and tracking in Appendix B.

7.1.4 Safety Considerations

Exploration, by definition, requires an autonomous system to deviate from known trajectories and venture into uncharted regions of the state space. This inherent divergence presents a critical challenge: ensuring operational safety during active exploration. In real-world deployment, a robot must strictly respect physical boundaries to prevent catastrophic damage to itself, the environment, or nearby humans. Because the methods and algorithms developed in this thesis operate natively within the task space and are purely geometric in nature, they handle safety through a two-tier paradigm, each carrying specific trade-offs.

First, our diffusion-based framework implicitly accounts for environment-level safety by directly encoding geometric constraints and obstacles as zero-Neumann or Dirichlet boundary conditions within the governing diffusion processes. A limitation of this approach is that obstacles are modeled purely as static geometric barriers because it ignores the robot's actual system dynamics and inertia.

Second, because our architecture is fundamentally modular and dedicated to generating a reference trajectory rather than raw joint-level commands, it seamlessly decouples geometric planning from low-level safety enforcement. This modularity allows the geometry-aware reference paths to be directly coupled with downstream control-theoretic safety frameworks, such as control barrier functions or predictive safety filters. However, this decoupling introduces a secondary limitation: a fundamental optimization mismatch can occur if a strict downstream safety filter heavily overrides or truncates the commands to enforce physical constraints. While this cascade strictly guarantees physical safety, the resulting tracking deviations can degrade the planned multi-scale coverage properties, leading to a localized loss of ergodicity in highly constrained workspaces.

7.2 Future Work

7.2.1 Exploration and Adaptation of Variable Compliance Behaviors

Throughout this thesis, the stiffness and damping gains for compliant control were assumed to be static and manually tuned. In practical scenarios, however, optimal compliance parameters vary dynamically across different object surfaces, task phases, and environmental conditions. Parameterizing general task-space compliance gains as six-dimensional symmetric positive definite (SPD) matrices results in a high-dimensional, non-Euclidean parameter space that is notoriously difficult to explore efficiently.

To address this, active learning [95] of impedance parameters can be formulated as a coupled exploration and estimation problem constrained to the non-Euclidean manifold of SPD matrices (S^6_{++}). Given that the SPD manifold admits a well-defined heat kernel [6], the framework introduced in Chapter 4 can be extended to enable geometry-aware active learning of these parameters[1]. An inherent advantage of this formulation over traditional methods is its capacity to initialize exploration from a prior distribution over compliance parameters. This prior can encode expert knowledge, such as promoting lower stiffness for safety, or transfer values from previously learned similar tasks. Furthermore, such an approach leverages the smoothness and geometric structure of the SPD manifold to continuously explore the compliance parameter space, mitigating the instantaneous jumps and control instabilities associated with discrete parameter sampling. Still, implementing this framework introduces two challenges. First, evaluating system performance under varying compliance parameters requires a robust reward function that simultaneously captures task execution success and safety margins. Recent learning-based task progress estimation methods offer promising advancements in

[1]Notably, similar strategies can be deployed for manipulability ellipsoids.

this direction [58, 21]. Second, computing the heat kernel on the full SPD manifold typically relies on finite-dimensional feature approximations rather than explicit analytical eigenfunctions, which can limit real-time computational efficiency. To trade-off this limitation with decoupling translational and rotational stiffness blocks, the heat kernel can be approximated by decomposing the components of the stiffness matrix into rotation and diagonal matrices, subsequently diffusing these distinct components over SO(3) and $\mathbb{R}^3$, respectively[2]. From this perspective, identifying optimal compliance parameters reduces to finding the appropriate reference frame (the rotation matrix) and the corresponding directional scalings (the diagonal matrix).

This perspective unveils a compelling paradigm: representing compliance parameters within the local reference frames defined by the orientation fields developed in Chapter 6. Expressed in a global reference frame fixed to an object, optimal compliance parameters vary substantially as a function of object shape. Conversely, by mapping compliance gains into object-centric local frames, task-relevant compliance directions are effectively decoupled from object-shape variations, flattening the underlying optimization landscape. For instance, if compliance variations are driven solely by object geometry, the parameters remain perfectly constant within local frames. If compliance must vary across the surface due to task phases, one can specify key compliance parameters at sparse anchor points and leverage the diffusion process to smoothly interpolate the compliance field across the continuous surface by independently diffusing the decomposed rotation and diagonal components.

Unifying the methodologies of Chapter 4 and Chapter 6 can enable the active learning of compliance behaviors natively within local reference frames. This fundamentally reduces the optimization domain to a lower-dimensional, geometry-invariant space, facilitating real-time compliance adaptation in contact-rich tasks such as cleaning, peeling, and slicing. Furthermore, as demonstrated in Chapter 6, object-centric local reference frames can be combined with other geometric primitives inducing spherical or cylindrical symmetries. Integrating these primitive-aligned fields with object-centric frames provides a structured foundation for composing complex, multi-modal compliance behaviors required for sophisticated manipulation tasks such as scooping.

7.2.2 Scaling to Multi-Contact Coordinated Manipulation

Dexterous manipulation requires extending robot-object interactions from a localized, single tool center point (TCP) to distributed, multi-contact configurations across multiple kinematic chains. This requirement is foundational for bimanual systems, multi-fingered dexterous hands, and humanoid platforms. As demonstrated in our collaborative work [64], geometric primitives and their corresponding similarity transformations provide a compact, highly structured representation for the cooperative task spaces of multi-kinematic chain systems.

[2] If coupling between the rotational and translational components of a full 6×6 task-space stiffness matrix is required, this decomposition can be extended using the Lie algebra of similarity transformations [64].

Although cooperative task spaces explicitly model the robotic system, they lack an intrinsic representation of the object. Conversely the diffused fields presented in Chapter 6 provide a rich, object-centric task space but abstract the robotic system down to a single point.

A promising future direction is to unify cooperative and object-centric task spaces to enable coordinated, multi-contact manipulation of complex geometries. By embedding the cooperative task space within the object's diffused field, the target configuration of the multi-body robot at any point would be directly governed by the underlying object geometry. We envision two primary avenues for this integration:

- Complementary Augmentation: In a bimanual manipulation task, for example, the diffused orientation field defines localized, geometry-aligned reference frames for each end-effector, while a line primitive parameterizes the cooperative task space (e.g., the relative distance and alignment) between the two hands. Similarly, for a multi-fingered hand, the diffused orientation field establishes local reference frames at each fingertip, while spherical or planar primitives define the cooperative constraints bounding the grasp.
- Primitive Diffusion via Sim(3): Alternatively, the diffused orientation field can be extended into a diffused field of geometric primitives (such as lines, circles, planes, or spheres) by leveraging the Lie algebra of similarity transformations ($\mathfrak{sim}(3)$) instead of the unit quaternion Lie algebra in Appendix B. Under this formulation, the spatial configuration of the cooperative task space varies continuously as a function of the object's geometry. Consequently, instead of defining key orientations, an operator can specify key geometric primitives at sparse anchor points and diffuse them across the surface and then the workspace. This yields a continuous field of primitives that can be directly queried by the multi-kinematic chain system for real-time coordinated control.

Formulating diffusion processes on the Lie algebra of similarity transformations also provides a mechanism to efficiently explore the space of geometric primitives, drastically reducing the dimensionality of the search space during coordinated manipulation. This dimensionality reduction stems from the fact that cooperative systems do not exploit their full, independent redundancy; rather, their motions are strictly constrained by a shared manipulation objective. The computational benefits of this geometric framework increase with the system's degrees of freedom, making it particularly useful for whole-body humanoid control and high-DOF dexterous hands.

7.2.3 Integration with Learning Frameworks

A powerful synthesis lies in integrating the proposed smooth geometric representations with learning-based frameworks for incorporating semantic understanding and generalization to different tasks and environments.

Using Learning Modules for Semantic Understanding

Integrating large Vision-Language Models (VLMs) offers a scalable avenue to infuse high-level semantic reasoning into our geometric architectures, replacing hand-engineered heuristics. As illustrated by our preliminary results in Appendix B, these foundation models can automatically initialize target segmentation masks [77], define target probability distributions for ergodic exploration, and anchor keypoints for diffusion fields.

For example, in a surface-cleaning task, a high-level open-vocabulary command such as "segment the dirt regions on kitchenware" can extract target distributions across novel object instances. A core advantage of this coupled architecture is that the downstream ergodic controller can act as a robust regularizer even if the VLM's semantic segmentation is noisy or imperfect. Similarly, task-space keypoints can be annotated on a single canonical template and transferred zero-shot to entirely different instances of the same object class encountered at runtime by leveraging dense semantic correspondence models such as DINOv2 [75].

Finally, in long-horizon manipulation scenarios, VLMs can be leveraged as high-level temporal sequencing engines. We envision a framework where a VLM reasons over a scene to dynamically sequence a series of geometric subgoals. By orchestrating a timeline of shifting segmentation masks and spatial keypoints, the model can guide the robot through different semantic phases of a complex task, such as transitioning from grasping a tool, to aligning it with an object, and ultimately executing a contact-rich manipulation skill.

Data Collection for Learning

Real-world interaction data remains a scarce, expensive, and inherently unsafe commodity to generate via unconstrained robotic exploration. While specialized teleoperation and demonstration interfaces [111] mitigate immediate safety risks, they lack scalability; furthermore, learning from passive human videos completely omits the critical tactile feedback and force-torque profiles necessary for contact-rich tasks.

A compelling avenue for future research is to develop an automated data collection framework that is warm-started by expert demonstrations and driven by an extension of the adaptive ergodic imitation framework presented in Chapter 5 (see Berrueta *et al.* [12] for foundations of ergodicity in statistical learning with embodied agents). In contrast to our online adaptation setting, where the robot actively conforms to novel environmental variations, this data collection pipeline would operate within a controlled, nominal environment. We envision a framework where the temperature parameter governing the diffusivity of the target distribution is systematically scheduled. Exploration would initiate with a low-variance policy closely tethered to the demonstration manifold. Over successive trials, the diffusion coefficient would be gradually inflated, forcing the system to progressively expand its state-space exploration until it encounters consistent task failure.

A primary technical challenge in implementing this self-limiting exploration sequence will be the continuous, automated assessment of task execution viability. Future implementations could leverage recent learning-based task progress estimators to evaluate these safety and performance margins online [58, 21], automatically halting or resetting the policy when an irrecoverable state boundary is crossed.

While this exploratory pipeline could be deployed simply to augment datasets for behavioral cloning, a more profound future direction lies in utilizing this highly diversified data to train a data-driven forward dynamics model, or world model. Traditional world models are notoriously vulnerable to compounding errors and exploitation outside the narrow expert data distribution. By systematically gathering data along the boundaries of failure, this framework might provide the data required to regularize the world model's state space. Investigating how such a boundary data can be used to capture the transition dynamics between successful compliance and task failure might be a an interesting direction toward learning world models capable of predicting and executing recovery behaviors.

Integration to Existing Policy Learning Frameworks

Building on our preliminary results, a promising direction for future research is to utilize the object-centric representation developed in Chapter 6 as a shape-invariant basis for imitation learning and reinforcement learning. In the context of imitation learning, this shape-invariant representation provides a unified basis to aggregate and align demonstrations performed across entirely distinct object shapes. A significant advantage of this formulation is that policies learned within this shape-invariant basis naturally generalize to previously unseen object geometries encountered at runtime. Analogously, this representation opens up a scalable pathway for sim-to-real transfer and curriculum learning. Because the underlying local reference frames can be computed across primitives, meshes and point clouds, policies can be trained initially on simple, idealized simulated geometries such as 2D primitives, progressed to watertight meshes in physics simulators, and ultimately transferred zero-shot to complex, irregular real-world objects perceived as raw, partial point clouds.

Rather than learning trajectories, a particularly compelling direction is to formulate the policy output as a sequence of sparse subgoals or keypoints defined natively within the diffused field. This design introduces a hierarchical control architecture that cleanly decouples high-level task planning from low-level geometric execution:

- High-Level Task Sequencing: The learning-based policy is unburdened from generating dense, continuous task-space trajectories. Instead, it operates at a lower control frequency, focusing entirely on high-level decision-making, semantic sequencing, and specifying discrete spatial subgoals parameterized as keypoints, keydirections or keyorientations.
- Low-Level Geometric Regularization: The continuous, underlying diffused field acts as a geometric dynamical system that handles trajectory generation on the fly. By treating

the policy's discrete subgoals as spatial guidelines, the field automatically synthesizes smooth, collision-avoiding, and geometry-adhering paths that strictly respect the object's geometry.

7.3 Broader Implications

Traditionally, robot manipulators have been designed with a central focus on precision and repeatability. While this paradigm excels in tightly controlled factory settings where every process is either fully modeled, measured ,or controlled, it breaks down under environmental variation. In contrast, humans operate under low sensorimotor precision, incomplete information, and imperfect models, yet adapt effortlessly to environmental variation. Crucially, rather than expecting single-trial success at a fixed horizon deployment or repeatedly executing a failing trajectory, humans rely on iterative, online adaptation to achieve task success asymptotically (Chapter 4 and 5). Looking forward, we argue that building robots for absolute precision, trying to acquire complete sensory information, and rigid trajectory tracking is often not only unrealistic, but inherently limiting in real-world deployments. This dissertation advocates for a fundamental paradigm shift toward variation-tolerant manipulation, driven by three core principles:

- Prioritising online sensor data over precision and completeness.
- Asymptotic success through adaptation instead of a fixed-horizon deployment.
- Specifying tasks using invariant structures rather than rigid trajectory tracking.

Moving forward, these principles offer important implications.

First, hardware and control requirements can be relaxed. Robots built with lower-precision sensorimotor components can still asymptotically succeed at high-precision tasks, even when trajectory tracking requirements exceed hardware specifications. As demonstrated in insertion ([87], Chapter 5) and cleaning tasks ([53], Chapter 4), online data can be regularized using appropriate inductive biases and used for building adaptive representations. By leveraging the task's invariant structure, control can be formulated as matching trajectory statistics asymptotically rather than strictly following a fixed-horizon trajectory. Although single-trial or fixed-horizon performance may initially be low, the robot can adapt iteratively and ultimately succeed due to the asymptotic coverage guarantees of ergodic control.

Second, adaptation should be an inherent property of the system, eliminating the need for additional task-transfer phases or data collection. For example, the surface cleaning objective in Chapter 2 is invariant to any specific trajectory, provided the trajectory's spatial statistics match the dirt distribution detected by the camera. Formulating the task via an ergodic controller yields seamless adaptation to varying dirt distributions and object geometries. Because measuring distributions and matching trajectory statistics across new shapes is trivial whereas trajectory transfer is challenging. Similarly, object-centric local reference frames in Chapter 4 can act as a shape-invariant basis for task encoding, enabling direct generalization across diverse object geometries.

Ultimately, we hope these insights broaden how the community approaches robot hardware design, task specification, data collection, and policy learning and transfer.

Bibliography

[1] Abraham, I., Prabhakar, A., Hartmann, M. J. Z., and Murphey, T. D. (2017). Ergodic Exploration using Binary Sensing for Non-Parametric Shape Estimation. *IEEE Robot. Autom. Lett.*, 2:827–834.

[2] Albini, A., Grella, F., Maiolino, P., and Cannata, G. (2021). Exploiting Distributed Tactile Sensors to Drive a Robot Arm Through Obstacles. *IEEE Robotics and Automation Letters*, 6:4361–4368.

[3] Amanhoud, W., Khoramshahi, M., and Billard, A. (2019). A Dynamical System Approach to Motion and Force Generation in Contact Tasks. In *Proc. Robotics: Science and Systems (RSS)*.

[4] Ayvali, E., Salman, H., and Choset, H. (2017). Ergodic coverage in constrained environments using stochastic trajectory optimization. In *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, pages 5204–5210.

[5] Ayvali, E., Srivatsan, R. A., Wang, L., Roy, R., Simaan, N., and Choset, H. (2016). Using Bayesian optimization to guide probing of a flexible environment for simultaneous registration and stiffness mapping. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 931–936.

[6] Azangulov, I., Smolensky, A., Terenin, A., and Borovitskiy, V. (2023). Stationary Kernels and Gaussian Processes on Lie Groups and their Homogeneous Spaces I: The compact case.

[7] Azangulov, I., Smolensky, A., Terenin, A., and Borovitskiy, V. (2024). Stationary Kernels and Gaussian Processes on Lie Groups and their Homogeneous Spaces II: Non-compact symmetric spaces.

[8] Barreiros, J., Beaulieu, A., Bhat, A., Cory, R., Cousineau, E., Dai, H., Fang, C.-H., Hashimoto, K., Irshad, M. Z., Itkina, M., Kuppuswamy, N., Lee, K.-H., Liu, K., McConachie, D., McMahon, I., Nishimura, H., Phillips-Grafflin, C., Richter, C., Shah, P., Srinivasan, K., Wulfe, B., Xu, C., Zhang, M., Alspach, A., Angeles, M., Arora, K., Guizilini, V. C., Castro, A., Chen, D., Chu, T.-S., Creasey, S., Curtis, S., Denitto, R., Dixon, E., Dusel, E., Ferreira, M., Goncalves, A., Gould, G., Guoy, D., Gupta, S., Han, X., Hatch, K., Hathaway, B., Henry, A., Hochsztein, H., Horgan, P., Iwase, S., Jackson, D., Karamcheti, S., Keh, S., Masterjohn, J., Masuda, M., Mercat, J.,

Miller, P., Mitiguy, P., Nguyen, T., Nimmer, J., Noguchi, Y., Ong, R., Onol, A., Pfannenstiehl, O., Poyner, R., Rocha, L. P. M., Richardson, G., Rodriguez, C., Seale, D., Sherman, M., Smith-Jones, M., Tago, D., Tokmakov, P., Tran, M., Hoorick, B. V., Vasiljevic, I., Zakharov, S., Zolotas, M., Ambruș, R., Fetzer-Borelli, K., Burchfiel, B., Kress-Gazit, H., Feng, S., Ford, S., and Tedrake, R. (2026). A careful examination of large behavior models for multitask dexterous manipulation. *Science RoBoticS*.

[9] Beber, L., Bilaloglu, C., Lamon, E., Saveriano, M., Fontanelli, D., Palopoli, L., and Calinon, S. (2026). Ergodic Palpation on Deformable Curved Surfaces for Viscoelastic Imaging and Lesion Localisation.

[10] Beber, L., Lamon, E., Moretti, G., Saveriano, M., Fambri, L., Palopoli, L., and Fontanelli, D. (2025). Force-Based Viscosity and Elasticity Measurements for Material Biomechanical Characterization With a Collaborative Robotic Arm. *IEEE Transactions on Instrumentation and Measurement*, 74:1–14.

[11] Belkin, M., Sun, J., and Wang, Y. (2009). Constructing Laplace Operator from Point Clouds in R^d. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1031–1040.

[12] Berrueta, T. A., Pinosky, A., and Murphey, T. D. (2024). Maximum diffusion reinforcement learning. *Nature Machine Intelligence*, 6:504–514.

[13] Bilaloglu, C., Löw, T., and Calinon, S. (2023). Whole-Body Ergodic Exploration With a Manipulator Using Diffusion. *IEEE Robotics and Automation Letters*, 8:8581–8587.

[14] Bilaloglu, C., Löw, T., and Calinon, S. (2025). Tactile Ergodic Coverage on Curved Surfaces. *IEEE Transactions on Robotics*, 41:1421–1435.

[15] Billard, A., Albu-Schaeffer, A., Beetz, M., Burgard, W., Corke, P., Ciocarlie, M., Dahiya, R., Kragic, D., Goldberg, K., Nagai, Y., and Scaramuzza, D. (2025). A roadmap for AI in robotics. *Nature Machine Intelligence*, 7:818–824.

[16] Borovitskiy, V., Terenin, A., Mostowsky, P., and Deisenroth, M. P. (2023). Matérn Gaussian processes on Riemannian manifolds.

[17] Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2017). Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34:18–42.

[18] Calinon, S. (2016). A tutorial on task-parameterized movement learning and retrieval. *Intelligent Service Robotics*, 9:1–29.

[19] Calli, B., Singh, A., Walsman, A., Srinivasa, S., Abbeel, P., and Dollar, A. M. (2015). The YCB object and Model set: Towards common benchmarks for manipulation research. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 510–517.

[20] Cao, J., Tagliasacchi, A., Olson, M., Zhang, H., and Su, Z. (2010). Point Cloud Skeletons via Laplacian Based Contraction. In *2010 Shape Modeling International Conference*, pages 187–197.

[21] Chen, Q., Yu, J., Schwager, M., Abbeel, P., Shentu, Y., and Wu, P. (2026). SARM: Stage-Aware Reward Modeling for Long Horizon Robot Manipulation.

[22] Chen, T., Cousineau, E., Kuppuswamy, N., and Agrawal, P. (2025). Vegetable Peeling: A Case Study in Constrained Dexterous Manipulation. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*.

[23] Chi, C., Feng, S., Du, Y., Xu, Z., Cousineau, E., Burchfiel, B., and Song, S. (2023). Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proc. Robotics: Science and Systems (RSS)*.

[24] Crane, K., De Goes, F., Desbrun, M., and Schröder, P. (2013a). Digital geometry processing with discrete exterior calculus. In *ACM SIGGRAPH 2013 Courses*, pages 1–126.

[25] Crane, K., Pinkall, U., and Schröder, P. (2013b). Robust fairing via conformal curvature flow. *ACM Transactions on Graphics*, 32:1–10.

[26] Crane, K., Weischedel, C., and Wardetzky, M. (2013c). Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Transactions on Graphics*, 32:1–11.

[27] Crnković, B., Ivić, S., and Zovko, M. (2023). Fast algorithm for centralized multi-agent maze exploration.

[28] Dong, C., Yu, L., Takizawa, M., Kudoh, S., and Suehiro, T. (2021). Food Peeling Method for Dual-arm Cooking Robot. In *2021 IEEE/SICE International Symposium on System Integration (SII)*, pages 801–806.

[29] Feng, N. and Crane, K. (2024). A Heat Method for Generalized Signed Distance. *ACM Transactions on Graphics*, 43:1–19.

[30] Feragen, A., Lauze, F., and Hauberg, S. (2014). Geodesic Exponential Kernels: When Curvature and Linearity Conflict.

[31] Gao, J., Jin, X., Krebs, F., Jaquier, N., and Asfour, T. (2024). Bi-KVIL: Keypoints-based Visual Imitation Learning of Bimanual Manipulation Tasks. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 16850–16857.

[32] Gao, J., Tao, Z., Jaquier, N., and Asfour, T. (2023). K-VIL: Keypoints-Based Visual Imitation Learning. *IEEE Transactions on Robotics*, 39:3888–3908.

[33] Gao, W. and Tedrake, R. (2021). kPAM 2.0: Feedback Control for Category-Level Robotic Manipulation. *IEEE Robotics and Automation Letters*, 6:2962–2969.

[34] Gilbert, E., Johnson, D., and Keerthi, S. (1988). A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal on Robotics and Automation*, 4:193–203.

[35] Gong, Y., Laha, R., and Figueredo, L. (2025). GeoPF: Infusing Geometry Into Potential Fields for Reactive Planning in Non-Trivial Environments. *IEEE Robotics and Automation Letters*, 10:11546–11553.

[36] Grambow, N., Fenner, L.-M., Kempkes, F., Hotz, P., Wan, D., Krüger, J., and Haninger, K. (2026). Anomaly detection for generic failure monitoring in robotic assembly, screwing and manipulation. *IEEE Robotics and Automation Letters*, 11:4737–4744.

[37] He, C., Liu, X., Camps, G. S., Sartoretti, G., and Schwager, M. (2025). Demystifying Diffusion Policies: Action Memorization and Simple Lookup Table Alternatives.

[38] Ho, J., Jain, A., and Abbeel, P. (2020). Denoising Diffusion Probabilistic Models.

[39] Hou, Y., Liu, Z., Chi, C., Cousineau, E., Kuppuswamy, N., Feng, S., Burchfiel, B., and Song, S. (2025). Adaptive Compliance Policy: Learning Approximate Compliance for Diffusion Guided Control. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*.

[40] Hughes, C., Warren, H., Lee, D., Ramos, F., and Abraham, I. (2025). Ergodic Trajectory Optimization on Generalized Domains Using Maximum Mean Discrepancy. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 01–07.

[41] Intelligence, P., Ai, B., Amin, A., Aniceto, R., Balakrishna, A., Balke, G., Black, K., Bokinsky, G., Cao, S., Charbonnier, T., Choudhary, V., Collins, F., Conley, K., Connors, G., Darpinian, J., Dhabalia, K., Dhaka, M., DiCarlo, J., Driess, D., Equi, M., Esmail, A., Fang, Y., Finn, C., Glossop, C., Godden, T., Goryachev, I., Groom, L., Habeeb, H., Hancock, H., Hausman, K., Hussein, G., Hwang, V., Ichter, B., Jacobsen, C., Jakubczak, S., Jen, R., Jones, T., Kammerer, G., Katz, B., Ke, L., Khadikov, M., Kuchi, C., Lamb, M., LeBlanc, D., LeCount, B., Levine, S., Li, X., Li-Bell, A., Lialin, V., Liang, Z., Lim, W., Lu, Y., Luo, E., Mano, V., Marwaha, N., Mongush, A., Murphy, L., Nair, S., Patterson, T., Pertsch, K., Ren, A. Z., Schelske, G., Sharma, C., Shi, B., Shi, L. X., Smith, L., Springenberg, J. T., Stachowicz, K., Stoeckle, W., Tang, J., Tanner, J., Tekeste, S., Torne, M., Vedder, K., Vuong, Q., Walling, A., Wang, H., Wang, J., Wang, X., Whalen, C., Whitmore, S., Williams, B., Xu, C., Yoo, S., Yu, L., Zhang, W., Zhang, Z., and Zhilinsky, U. (2026). \$$\pi$_{0.7}\$: A Steerable Generalist Robotic Foundation Model with Emergent Capabilities.

[42] Ivić, S. (2022). Motion Control for Autonomous Heterogeneous Multiagent Area Search in Uncertain Conditions. *IEEE Transactions on Cybernetics*, 52:3123–3135.

[43] Ivić, S., Crnković, B., Grbčić, L., and Matleković, L. (2023). Multi-UAV trajectory planning for 3D visual inspection of complex structures. *Automation in Construction*, 147:104709.

[44] Ivić, S., Crnković, B., and Mezić, I. (2017). Ergodicity-Based Cooperative Multiagent Area Coverage via a Potential Field. *IEEE Transactions on Cybernetics*, 47:1983–1993.

[45] Ivić, S., Sikirica, A., and Crnković, B. (2022). Constrained multi-agent ergodic area surveying control based on finite element approximation of the potential field. *Engineering Applications of Artificial Intelligence*, 116:105441.

[46] Jacobs, H., Nair, S., and Marsden, J. (2010). Multiscale surveillance of Riemannian manifolds. In *Proceedings of the 2010 American Control Conference*, pages 5732–5737.

[47] Jaquier, N., Welle, M. C., Gams, A., Yao, K., Fichera, B., Billard, A., Ude, A., Asfour, T., and Kragic, D. (2025). Transfer learning in robotics: An upcoming breakthrough? A review of promises and challenges. *The International Journal of Robotics Research*, 44:465–485.

[48] Jin, Y., Lv, J., Yu, W., Fang, H., Li, Y.-L., and Lu, C. (2025). SIME: Enhancing Policy Self-Improvement with Modal-level Exploration.

[49] Jones, I. and Lanners, D. (2026). Computing Diffusion Geometry.

[50] Kabir, A. M., Kaipa, K. N., Marvel, J., and Gupta, S. K. (2017). Automated Planning for Robotic Cleaning Using Multiple Setups and Oscillatory Tool Motions. *IEEE Transactions on Automation Science and Engineering*, 14:1364–1377.

[51] Kalakrishnan, M., Chitta, S., Theodorou, E., Pastor, P., and Schaal, S. (2011). STOMP: Stochastic trajectory optimization for motion planning. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 4569–4574.

[52] Kalinowska, A., Prabhakar, A., Fitzsimons, K., and Murphey, T. (2021a). Ergodic imitation: Learning from what to do and what not to do.

[53] Kalinowska, A., Prabhakar, A., Fitzsimons, K., and Murphey, T. (2021b). Ergodic imitation: Learning from what to do and what not to do. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 3648–3654.

[54] Karacan, K., Sadeghian, H., Kirschner, R., and Haddadin, S. (2022). Passivity-Based Skill Motion Learning in Stiffness-Adaptive Unified Force-Impedance Control. In *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, pages 9604–9611.

[55] Ketchum, J., Prabhakar, A., and Murphey, T. D. (2024). Active Exploration for Real-Time Haptic Training. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 9689–9695.

[56] Lee, D., Lerch, C., Ramos, F., and Abraham, I. (2024). Stein Variational Ergodic Search.

[57] Li, Y., Darwiche, N., Razmjoo, A., Liu, S., Du, Y., Ijspeert, A., and Calinon, S. (2025). Geometry-aware Policy Imitation.

[58] Liang, A., Korkmaz, Y., Zhang, J., Hwang, M., Anwar, A., Kaushik, S., Shah, A., Huang, A. S., Zettlemoyer, L., Fox, D., Xiang, Y., Li, A., Bobu, A., Gupta, A., Tu, S., Biyik, E., and Zhang, J. (2026). Robometer: Scaling General-Purpose Robotic Reward Models via Trajectory Comparisons.

[59] Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. (2023). Flow Matching for Generative Modeling.

[60] Lipman, Y., Rustamov, R. M., and Funkhouser, T. A. (2010). Biharmonic distance. *ACM Transactions on Graphics*, 29:1–11.

[61] Liu, Q. and Wang, D. (2016). Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm.

[62] Liu, W., Wang, J., Wang, Y., Wang, W., and Lu, C. (2025). ForceMimic: Force-Centric Imitation Learning with Force-Motion Capture System for Contact-Rich Manipulation. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*.

[63] Liu, Y., Prabhakaran, B., and Guo, X. (2012). Point-Based Manifold Harmonics. *IEEE Transactions on Visualization and Computer Graphics*, 18:1693–1703.

[64] Löw, T., Bilaloglu, C., and Calinon, S. (2025). Cooperative Task Spaces for Multi-Arm Manipulation Control based on Similarity Transformations.

[65] Löw, T. and Calinon, S. (2023). Geometric Algebra for Optimal Control with Applications in Manipulation Tasks. *IEEE Transactions on Robotics*, 39:3586–3600.

[66] Manuelli, L., Gao, W., Florence, P., and Tedrake, R. (2019). kPAM: KeyPoint Affordances for Category-Level Robotic Manipulation. In *The International Symposium on Robotic Research*.

[67] Markley, F. L., Cheng, Y., Crassidis, J. L., and Oshman, Y. (2007). Averaging Quaternions. *Journal of Guidance, Control, and Dynamics*, 30:1193–1197.

[68] Mathew, G. and Mezic, I. (2009). Spectral Multiscale Coverage: A uniform coverage algorithm for mobile sensor networks. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC)*, pages 7872–7877.

[69] Mathew, G. and Mezić, I. (2011). Metrics for ergodicity and design of ergodic dynamics for multi-agent systems. *Physica D: Nonlinear Phenomena*, 240:432–442.

[70] Miller, B., Sawhney, R., Crane, K., and Gkioulekas, I. (2024). Differential Walk on Spheres. *ACM Transactions on Graphics*, pages 1–18.

[71] Miller, L. M., Silverman, Y., MacIver, M. A., and Murphey, T. D. (2016). Ergodic Exploration of Distributed Information. *IEEE Transactions on Robotics*, 32:36–52.

[72] Mittendorfer, P. and Cheng, G. (2011). Humanoid Multimodal Tactile-Sensing Modules. *IEEE Transactions on Robotics*, 27:401–410.

[73] Muchacho, R. I. C. and Pokorny, F. T. (2024). Walk on Spheres for PDE-based Path Planning. In *arXiv:2406.01713 [Cs.RO]*. arXiv:2406.01713 [cs.RO].

[74] Nealen, A. (2004). An As-Short-As-Possible Introduction to the Least Squares, Weighted Least Squares and Moving Least Squares Methods for Scattered Data Approximation and Interpolation.

[75] Oquab, M., Darcet, T., Moutakanni, T., Vo, H. V., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.-Y., Li, S.-W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., and Bojanowski, P. (2024). DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research.*

[76] Razmjoo, A., Calinon, S., Gienger, M., and Zhang, F. (2025). CCDP: Composition of Conditional Diffusion Policies with Guided Sampling.

[77] Ren, T., Liu, S., Zeng, A., Lin, J., Li, K., Cao, H., Chen, J., Huang, X., Chen, Y., Yan, F., Zeng, Z., Zhang, H., Li, F., Yang, J., Li, H., Jiang, Q., and Zhang, L. (2024). Grounded SAM: Assembling Open-World Models for Diverse Visual Tasks. arXiv:2401.14159 [cs.CV].

[78] Roberts, G. O. and Tweedie, R. L. (1996). Exponential Convergence of Langevin Distributions and Their Discrete Approximations. *Bernoulli*, 2:341.

[79] Sawhney, R. and Crane, K. (2020). Monte Carlo Geometry Processing:A Grid-Free Approach to PDE-Based Methods on Volumetric Domains. *ACM Transactions on Graphics*, 38:1–18.

[80] Scherzinger, S., Roennau, A., and Dillmann, R. (2017). Forward Dynamics Compliance Control (FDCC): A new approach to cartesian compliance for robotic manipulators. In *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, pages 4568–4575.

[81] Schulman, J., Duan, Y., Ho, J., Lee, A., Awwal, I., Bradlow, H., Pan, J., Patil, S., Goldberg, K., and Abbeel, P. (2014). Motion planning with sequential convex optimization and convex collision checking. *The International Journal of Robotics Research*, 33:1251–1270.

[82] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal Policy Optimization Algorithms. arXiv:1707.06347 [cs.LG].

[83] Sharp, N., Attaiki, S., Crane, K., and Ovsjanikov, M. (2022). DiffusionNet: Discretization Agnostic Learning on Surfaces. *ACM Transactions on Graphics*, 41:1–16.

[84] Sharp, N. and Crane, K. (2020). A Laplacian for Nonmanifold Triangle Meshes. *Computer Graphics Forum*, 39:69–80.

[85] Sharp, N., Soliman, Y., and Crane, K. (2019). The Vector Heat Method. *ACM Transactions on Graphics*, 38:1–19.

[86] Shaw, K., Agarwal, A., and Pathak, D. (2023). LEAP Hand: Low-Cost, Efficient, and Anthropomorphic Hand for Robot Learning. In *Proc. Robotics: Science and Systems (RSS).*

[87] Shetty, S., Silvério, J., and Calinon, S. (2022). Ergodic Exploration Using Tensor Train: Applications in Insertion Tasks. *IEEE Transactions on Robotics*, 38:906–921.

[88] Shorthose, O., Albini, A., He, L., and Maiolino, P. (2022). Design of a 3D-Printed Soft Robotic Hand With Integrated Distributed Tactile Sensing. *IEEE Robotics and Automation Letters*, 7:3945–3952.

[89] Simeonov, A., Du, Y., Tagliasacchi, A., Tenenbaum, J. B., Rodriguez, A., Agrawal, P., and Sitzmann, V. (2022a). Neural Descriptor Fields: SE(3)-Equivariant Object Representations for Manipulation. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 6394–6400.

[90] Simeonov, A., Du, Y., Yen-Chen, L., Rodriguez, A., Kaelbling, L. P., Lozano-Perez, T., and Agrawal, P. (2022b). SE(3)-Equivariant Relational Rearrangement with Neural Descriptor Fields. In *Proceedings of The Conference on Robot Learning (CoRL)*, pages 835–846.

[91] So, P., Cabral Muchacho, R. I., Jeanne Kirschner, R., Swikir, A., Figueredo, L., Abu-Dakka, F. J., and Haddadin, S. (2024). CITR: A Coordinate-Invariant Task Representation for Robotic Manipulation. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, pages 17501–17507.

[92] Spahn, M., Wisse, M., and Alonso-Mora, J. (2023). Dynamic Optimization Fabrics for Motion Generation. *IEEE Transactions on Robotics*, pages 1–16.

[93] Sun, M., Gaggar, A., Trautman, P., and Murphey, T. (2024). Fast Ergodic Search with Kernel Functions.

[94] Sun, M. M., Pinosky, A., and Murphey, T. (2025). Flow Matching Ergodic Coverage.

[95] Taylor, A. T., Berrueta, T. A., and Murphey, T. D. (2021). Active learning in robotics: A review of control principles. *Mechatronics*, 77:102576.

[96] Ti, B., Razmjoo, A., Gao, Y., Zhao, J., and Calinon, S. (2023). A geometric optimal control approach for imitation and generalization of manipulation skills. *Robotics and Autonomous Systems*, 164.

[97] Ureche, A. L. P., Umezawa, K., Nakamura, Y., and Billard, A. (2015). Task Parameterization Using Continuous Constraints Extracted From Human Demonstrations. *IEEE Transactions on Robotics*, 31:1458–1471.

[98] Van Wyk, K., Xie, M., Li, A., Rana, M. A., Babich, B., Peele, B., Wan, Q., Akinola, I., Sundaralingam, B., Fox, D., Boots, B., and Ratliff, N. D. (2022). Geometric Fabrics: Generalizing Classical Mechanics to Capture the Physics of Behavior. *IEEE Robotics and Automation Letters*, 7:3202–3209.

[99] Varadhan, S. R. S. (1967). On the behavior of the fundamental solution of the heat equation with variable coefficients. *Communications on Pure and Applied Mathematics*, 20:431–455.

[100] Vochten, M., De Laet, T., and De Schutter, J. (2019). Generalizing demonstrated motion trajectories using coordinate-free shape descriptors. *Robotics and Autonomous Systems*, 122.

[101] Vochten, M., Mohammadi, A. M., Verduyn, A., De Laet, T., Aertbeliën, E., and De Schutter, J. (2023). Invariant Descriptors of Motion and Force Trajectories for Interpreting Object Manipulation Tasks in Contact. *IEEE Transactions on Robotics*, 39:4892–4912.

[102] Wang, Y., Zhang, M., Li, Z., Kelestemur, T., Driggs-Campbell, K., Wu, J., Fei-Fei, L., and Li, Y. (2024). D^{3}Fields: Dynamic 3D Descriptor Fields for Zero-Shot Generalizable Rearrangement. In *Proceedings of The Conference on Robot Learning (CoRL)*, pages 272–298.

[103] Xue, H., Ren, J., Chen, W., Zhang, G., Fang, Y., Gu, G., Xu, H., and Lu, C. (2025a). Reactive Diffusion Policy: Slow-Fast Visual-Tactile Policy Learning for Contact-Rich Manipulation. In *Proc. Robotics: Science and Systems (RSS)*.

[104] Xue, Z., Deng, S., Chen, Z., Wang, Y., Yuan, Z., and Xu, H. (2025b). DemoGen: Synthetic Demonstration Generation for Data-Efficient Visuomotor Policy Learning.

[105] Yan, Y. and Pan, J. (2021). Fast Localization and Segmentation of Tissue Abnormalities by Autonomous Robotic Palpation. *IEEE Robot. Autom. Lett.*, 6:1707–1714.

[106] Yang, J., Cao, Z.-a., Deng, C., Antonova, R., Song, S., and Bohg, J. (2024). EquiBot: SIM(3)-Equivariant Diffusion Policy for Generalizable and Data Efficient Learning.

[107] Ye, R., Hu, Y., Yuhan, Bian, Kulm, L., and Bhattacharjee, T. (2024). MORPHeus: A Multimodal One-armed Robot-assisted Peeling System with Human Users In-the-loop. In *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*.

[108] Yoshikawa, T. (1985). Manipulability of Robotic Mechanisms. *The International Journal of Robotics Research*, 4:3–9.

[109] Ze, Y., Zhang, G., Zhang, K., Hu, C., Wang, M., and Xu, H. (2024). 3D Diffusion Policy: Generalizable Visuomotor Policy Learning via Simple 3D Representations. In *Proc. Robotics: Science and Systems (RSS)*.

[110] Zhang, Z. and She, Y. (2026). EquiForm: Noise-Robust SE(3)-Equivariant Policy Learning from 3D Point Clouds.

[111] Zhao, T. Z., Kumar, V., Levine, S., and Finn, C. (2023). Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware.

[112] Zhu, X., Wang, F., Walters, R., and Shi, J. (2025). SE(3)-Equivariant Diffusion Policy in Spherical Fourier Space.

[113] Zucker, M., Ratliff, N., Dragan, A. D., Pivtoraiko, M., Klingensmith, M., Dellin, C. M., Bagnell, J. A., and Srinivasa, S. S. (2013). CHOMP: Covariant Hamiltonian optimization for motion planning. *The International Journal of Robotics Research*, 32:1164–1193.

A Appendix to Chapter 2

A.1 Tactile Ergodic Control Experimental and Implementation Parameters

This appendix reports all controller, estimation, and mapping hyperparameters used in the physical experiments.

A.1.1 Force–Impedance Control

Cartesian end–effector impedance parameters are set to

$$\mathbf{K}_{ee} = [800, 800, 0, 60, 60, 30].$$

Table A.1: Force–impedance and estimator parameters.

Parameter	**Hemisphere**	**Gaus. Dimple**
Normal force bias [N]	3	4.25
Force amplitude [N]	1.75	2.75
Excitation frequency	4π rad/s	4π rad/s
PID gains	[1, 3]	[1, 3]
$\alpha_{\max}$	1.003	1.003
θ	1.9	1.9

A.1.2 Gaussian Process Mapping

The intrinsic Gaussian Process (GP) used for viscoelastic mapping adopts the hyperparameters listed in Table A.2.

The prior mean is offset from the minimum registered stiffness $k_{\min}$ to avoid spurious peaks during the early exploration phase.

Table A.2: Gaussian Process hyperparameters and PC initialization.

Parameter	**Value**
Lengthscale l	0.005 m (indenter diameter)
Signal variance σ	1
Prior mean m	$k_{\text{min}} + 0.1\, k_{\text{min}}$
Geometry	**Voxel size [m]**
Hemisphere	2.0×10^{-3}
Concave	1.5×10^{-3}

B Appendix to Chapter 3

B.1 Supplementary Methods

B.1.1 Keypoint Extraction

Keypoints are inputs to our representation and can be obtained by automatic extraction, transfer with a learning based approach, or manual annotation. As shown in Fig. 6E, our method is robust to noise in keypoint extraction. In real-world experiments we used automatic keypoint extraction. As a proof-of-concept we also performed keypoint transfer using pretrained vision encoders [75].

Automatic Keypoint Extraction

For automatic keypoint extraction we assume the observed point cloud has an open boundary, which is the typical case for single camera acquisition.

For slicing and peeling experiments, the keypoints are the antipodal poles of the object, namely the stem end and the tip. We detect the two ends of elongated fruits and vegetables such as banana, pear, or cucumber with an iterative furthest point strategy on the point cloud driven by the scalar diffusion equation (Equation 2.10). If the point cloud is not elongated, the method may be unreliable and we recommend manual annotation.

First, we compute the geometric center of the surface and use it as a source similar to Equation 6.2. Next, we solve Equation 2.15 to diffuse a scalar field over the surface. The vertex with the minimum diffused value is selected as the first end point since it lies far along the intrinsic geodesic from the center. Then, we use the first detected end point as the new source and repeat the process and select the minimum value vertex as the second end point.

On a set of fifty randomly deformed banana point clouds with combined scaling, twisting, and bending, the two stage diffusion consistently identified anatomically correct ends at the stem and the tip (Fig. B.1A, i). The method requires no training data and transfers to other

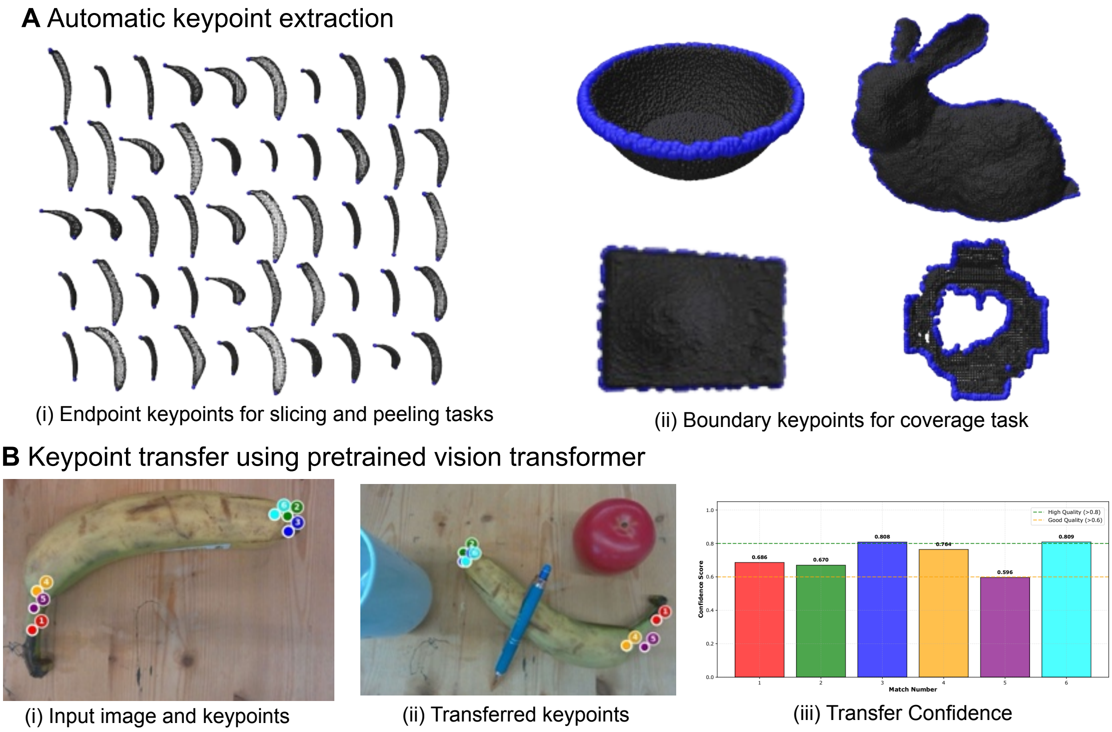


Figure B.1: **Keypoint extraction**. (**A**) Automatic keypoint extraction for slicing, peeling and coverage experiments. Point cloud is in black and keypoints are in blue. (i) 50 twisted, bent and anisotropically scaled instances of banana point cloud used for testing our automatic keypoint extraction routine. For slicing and peeling tasks, keypoints need to be at two ends of the object. (ii) Various real-world point clouds tested with our boundary keypoint extraction routine. For coverage tasks, the keypoints need to span the object boundary. (**B**) Keypoint transfer using pretrained vision transformer [75]. (i) User annotated keypoints on the input image. (ii) Zero-shot transfer of annotated keypoints to the image collected at runtime. (iii) Transfer confidence for each keypoint.

elongated objects since it relies only on intrinsic geometry.

In coverage experiments, we set the object boundary as the keypoints. Accordingly, one can use any boundary estimation procedure. We use a procedure inspired by the Point Cloud Library's boundaryEstimation function. We tested this approach with point clouds that we collected from various objects (Fig. B.1A, ii).

Keypoint Transfer

We used a pretrained vision encoder DINOv2 [75] to transfer keypoints annotated on one object to a target object encountered at runtime. This method requires no training, fine-tuning, or dataset-specific optimization. It relies entirely on the general-purpose visual understanding encoded in pretrained foundation models.

The core principle is that semantically similar image regions should produce similar feature representations, even under substantial appearance variations. For each query keypoint in the source image, we extract a local feature descriptor by processing a surrounding image patch through the DINOv2 encoder. We then search the target image for the location that maximizes feature similarity, measured via cosine similarity in the learned feature space. We provided the results of our proof-of-concept experiment in Fig. B.1B.

B.1.2 Local Policy Learning

Local action primitives can also be learned as local policies with reinforcement learning. In our experiments, we represented the policy using a multi-layer perceptron that was trained using proximal policy optimization (PPO) [82]. During training we used the following dense reward function

$$\begin{aligned}
r_t(\boldsymbol{s}_t, \boldsymbol{a}_t) = & f_1(\boldsymbol{s}_t, \boldsymbol{a}_t) + f_2(\boldsymbol{s}_t, \boldsymbol{a}_t) + f_3(\boldsymbol{s}_t, \boldsymbol{a}_t) + f_4(\boldsymbol{s}_t, \boldsymbol{a}_t) + f_5(\boldsymbol{s}_t, \boldsymbol{a}_t) \\
& f_1(\boldsymbol{s}_t, \boldsymbol{a}_t) = -0.1 \quad \text{time penalty (accumulating)} \\
& f_2(\boldsymbol{s}_t, \boldsymbol{a}_t) = -0.01\|\boldsymbol{a}\|^2 \quad \text{action penalty} \\
& f_3(\boldsymbol{s}_t, \boldsymbol{a}_t) = \begin{cases} 10(d_c(\boldsymbol{s}_{t-1}) - d_c(\boldsymbol{s}_t)) & \text{off circle} \\ 0.5 + 10(d_t(\boldsymbol{s}_{t-1}) - d_t(\boldsymbol{s}_t)) & \text{on circle} \end{cases} \\
& f_4(\boldsymbol{s}_t, \boldsymbol{a}_t) = -5d_c(\boldsymbol{s}_t) \quad \text{penetration penalty}
\end{aligned}, \tag{B.1}$$

where s_t and a_t are the states and actions at time t, $d_c(\cdot)$ is the distance to the circle and $d_t(\cdot)$ is the distance to the target. When the target is reached, we additionally give a single sparse reward

$$r_f(\boldsymbol{s}_t) = 10 \quad \text{target reached reward (one time)}. \tag{B.2}$$

To speed up training, we pretrain with 10 demonstrations, then train the policy for 5×10^5 timesteps with the entropy coefficient annealed linearly from 0.04 to 0.002, which encourages initial exploration and convergence to a near deterministic policy. For the local policy, we compute the diffused orientation field using a keypoint at the target position.

B.1.3 Diffused Orientation Fields from Directed and Oriented Keypoints

Complementary to the scalar diffusion method described in the main text for computing diffused orientation field, we provide two alternatives that utilize directed and oriented keypoints.

Orientation Fields via Diffused Directed Keypoints

Solving scalar diffusion and computing its gradient yields directions that always points towards or away from the keypoints. However, in some applications, we might directly have access to direction information at keypoints that do not necessarily correspond to sources or sinks. For example, in a wiping task, we might want to specify the desired wiping direction at specific locations on the object surface and smoothly interpolate these directions.

Let $\Omega \subset \mathbb{R}^n$ and a vector field $\mathbf{v}(\boldsymbol{x}, \tau) = (v_1, v_2, \dots v_m)$. One can compute a diffused vector field by solving m independent scalar diffusion problems

$$\dot{v}_i = \Delta v_i \quad \text{in } \Omega, \qquad v_i\big|_{\partial\Omega} = g_i(\boldsymbol{x}), \qquad i = \{1, 2, \dots, m\}. \tag{B.3}$$

where $g_i(\boldsymbol{x})$ is the i^{th} component of the directed keypoints. Since diffusion would result in magnitude decay and we are interested in direction vectors, we normalize the diffused vector field to obtain the direction field.

For vectors on curved surfaces, an additional constraint arises: vectors must reside in the tangent bundle, $\mathbf{v} \in \mathcal{T}\mathcal{M}$, which is the manifold formed by the union of all tangent spaces of the original manifold $\mathcal{M}$. Consequently, the components of a vector field cannot be diffused independently, as curved manifolds lack a global coordinate system, and vectors at different points cannot be directly compared. To address this, one can use the connection Laplacian $\Delta^{\nabla}_{\mathcal{M}}$, defined as the composition of the covariant derivative and its adjoint for diffusing a tangent vector field $\mathbf{v}(\boldsymbol{x}, \tau)$:

$$\dot{\mathbf{v}}(\boldsymbol{x}, \tau) = \Delta^{\nabla}_{\mathcal{M}} \mathbf{v}(\boldsymbol{x}, \tau). \tag{B.4}$$

Similar to the Euclidean setting, one needs to normalize the diffused vector field to obtain the direction field on the manifold. We refer the reader to [85] for more details on vector diffusion for curved surfaces.

Orientation Fields via Diffused Oriented Keypoints

Scalar and vector diffusion yield local reference frames where the z-axis aligns with the surface normal, and the x–y plane is tangent to the surface. Although this is often useful, some applications might require diffusing desired full 3D orientations rather than surface-constrained directions. We refer to the desired full 3D orientations as oriented keypoints. For example, one can specify the full desired tool orientation at specific locations on the object and use orientation diffusion to smoothly interpolate these orientations over the entire workspace.

Notably, in order to diffuse full 3D orientations over a surface, we cannot use the method that we presented in the main paper for workspace diffusion as the Walk on Spheres method is limited to Euclidean spaces and cannot be used on curved surfaces. Also we cannot use the scalar or vector diffusion methods presented in the previous sections since they cannot represent full 3D orientations and SO(3) is not a vector space. To diffuse full 3D orientations over a surface, we must choose a suitable representation. If we use rotation matrices to represent orientations, component-wise diffusion can violate the orthogonality and determinant constraints of SO(3). A better alternative is to diffuse columns of the rotation matrix which correspond to the axes of the local reference frames independently, as we discuss in more detail in the next section. However, this is still an approximation as we ignore the structure of SO(3) during the diffusion then correct it afterwards using re-orthonormalization.

When having a few oriented keypoints, a more principled approach is to utilize the Lie algebra $\mathfrak{spin}(3)$, which forms a vector space over $\mathbb{R}$ and can be represented using pure quaternions (quaternions with zero scalar part). The associated Lie group Spin(3), represented by unit quaternions, is a double cover of SO(3), meaning that $\mathbf{q}$ and $-\mathbf{q}$ correspond to the same rotation. The exponential and logarithmic maps between $\mathfrak{spin}(3)$ and Spin(3) are given as:

$$\exp(\mathbf{u}\phi) = \mathbf{q} = \cos(\phi) + \sin(\phi)\mathbf{u}, \tag{B.5}$$

$$\log(\mathbf{q}) = \mathbf{u}\phi = \arccos(w)\frac{\mathbf{u}}{|\mathbf{u}|}. \tag{B.6}$$

These maps allow computations to be performed in a linear vector space, enabling independent diffusion of each component, as we described in the previous section. However, similar to the vector diffusion case, one needs to compensate for the magnitude decay due to diffusion. Notably, we cannot just normalize the diffused pure quaternions since their magnitude is related to the rotation angle. As proposed in [85], the magnitude decay due to diffusion can be measured using an indicator function on the boundaries

$$\phi(\boldsymbol{x}, 0) = \begin{cases} 1, & \boldsymbol{x} \in \partial\Omega \\ 0, & \boldsymbol{x} \in \Omega \end{cases}, \tag{B.7}$$

and by diffusing the magnitude as a scalar field

$$u(\boldsymbol{x}, 0) = \begin{cases} \|\mathbf{v}(\boldsymbol{x}, \tau)\|, & \boldsymbol{x} \in \partial\Omega \\ 0, & \boldsymbol{x} \in \Omega \end{cases}. \tag{B.8}$$

Then, one can compensate for the magnitude decay of the diffused vector field $\mathbf{v}(\boldsymbol{x}, t)$

$$\bar{\mathbf{v}}(\boldsymbol{x}, \tau) = \frac{\mathbf{v}(\boldsymbol{x}, \tau)\, u(\boldsymbol{x}, \tau)}{\|\mathbf{v}(\boldsymbol{x}, \tau)\|\phi(\boldsymbol{x}, \tau)}. \tag{B.9}$$

After the diffusion, we map back the diffused pure quaternions to unit quaternions. Our algorithm for diffusing orientations begins by representing orientations as unit quaternions $\mathbf{q}$. First, we ensure that all quaternions are in the same hemisphere by flipping their signs if necessary. Next, we map each quaternion to a pure quaternion $\mathbf{u} \in \mathfrak{spin}(3)$ using the logarithmic map (Equation B.6). We then impose these pure quaternions as Dirichlet boundary conditions in the diffusion equation (Equation 2.10) and solve it independently for each component of the pure quaternions, following the procedure in Equation B.3. We also diffuse the indicator function in Equation B.7 and the magnitude function in Equation B.8, compensating for the magnitude decay using Equation B.9. Finally, we map the diffused results back to unit quaternions using the exponential map (Equation B.5). Note that we recommend this method only for diffusing a few oriented keypoints on the surface, and we advise using the method presented in the main paper for workspace diffusion.

Approximated Diffused Orientation Field using Vector Diffusion and Orthonormalization

In this section, we describe a simple approximation for computing diffused orientation fields on the workspace. We diffuse orthonormal direction vectors independently followed by orthonormalization. Let an orientation at a point be represented by a rotation matrix

$$\mathbf{R} = \begin{bmatrix} \mathbf{r}_x & \mathbf{r}_y & \mathbf{r}_z \end{bmatrix} \in \mathrm{SO}(3), \tag{B.10}$$

whose columns $\mathbf{r}_x, \mathbf{r}_y, \mathbf{r}_z \in \mathbb{R}^3$ encode the local x, y, z axes. In order to diffuse individual column vectors, we independently solve three scalar diffusion equations as presented in Equation 2.12 and normalize to unit magnitude. We repeat this process for all three column vectors and combine the results into a field of 3×3 matrices $\mathbf{M}(\boldsymbol{x})$.

Because these three problems are solved independently in a vector space, the result $\mathbf{M}(\boldsymbol{x})$ generally violates the constraints of SO(3):

$$\mathbf{M}(\boldsymbol{x})^\top \mathbf{M}(\boldsymbol{x}) \neq \mathbf{I} \quad \text{and} \quad \det \mathbf{M}(\boldsymbol{x}) \neq 1. \tag{B.11}$$

We illustrate violations of orthogonality in Fig. B.2A.

To convert the diffused matrix $\mathbf{M}(\boldsymbol{x})$ into a valid orientation we orthonormalize it by projecting onto SO(3). The projection that makes the smallest Frobenius norm change,

$$\mathbf{R}^\star(\mathbf{x}) \;=\; \arg\min_{\mathbf{R}\in \mathrm{SO}(3)} \|\mathbf{M}(\mathbf{x}) - \mathbf{R}\|_F, \tag{B.12}$$

is given by the polar decomposition computed via a thin singular value decomposition (SVD) $\mathbf{M} = \mathbf{U\Sigma V}^\top$. This symmetric orthogonalization treats all columns of $\mathbf{M}$ in a balanced way and yields an orthonormal frame with determinant one. A sequential alternative is the Gram-Schmidt process applied to the columns of $\mathbf{M}$. However, the latter approach is order dependent and would introduce bias toward the first column and distort the other columns. We compare orthonormalized vector diffusion using both SVD and Gram-Schmidt with fixed z-direction to the orientation diffusion (Fig. B.2B). For a quantitative comparison, we measured the smoothness of the resulting orientation fields using angular deviation between neighboring local x-directions. For each point on the grid visualized in Fig. B.2, we computed the angle of neighboring local x-directions in the xz-plane with respect to a common reference direction. We present the statistics of the angular deviation in Table B.1. As expected, we obtain the smoothest field using the orientation diffusion method presented in the main paper, since

Table B.1: **Orientation smoothness metrics across methods.** We report the average angular deviation, standard deviation of the angular deviation, and maximum angular deviation between neighboring local x directions. Note that the exact values can change slightly (< 1%) due to the randomness of the Walk on Spheres.

Method	**Average**	**Std. Dev.**	**Max.**
Orthonormalized Vector Diffusion (Z-fixed)	9.80°	21.34°	179.90°
Orthonormalized Vector Diffusion (SVD)	7.12°	12.11°	160.19°
Orientation Diffusion (Ours)	6.25°	10.41°	152.58°

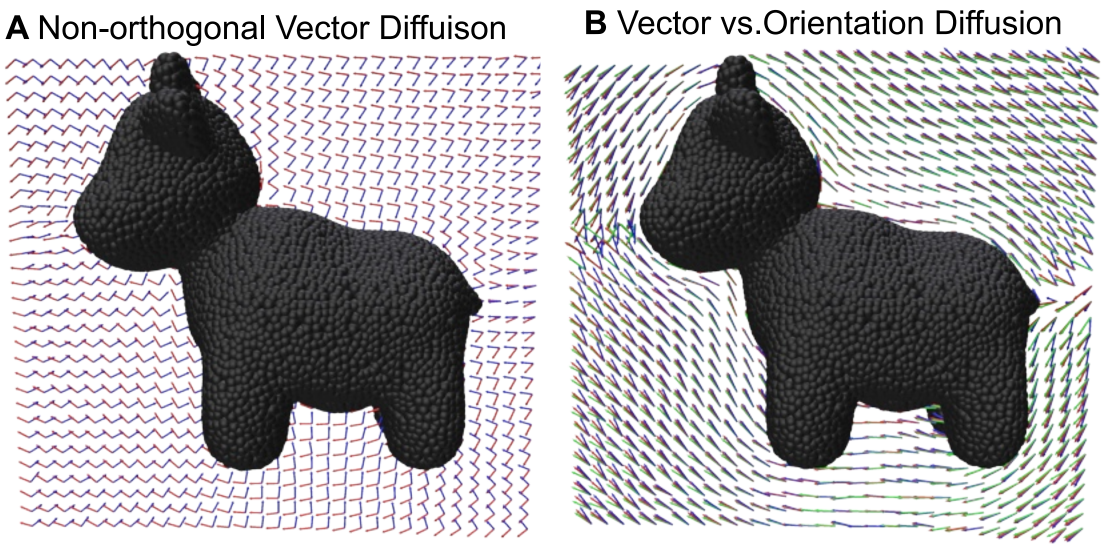


Figure B.2: **Diffusing the vectors composing the local frames independently**. (**A**) Diffusing vectors independently breaks orthogonality of orientations. (**B**) Comparison of the local reference frames computed by orthonormalized vector diffusion and orientation diffusion (our method). We visualize the x-axis of orientation diffusion in green, the x-axis of the orthonormalized vector diffusion using SVD orthogonalization in red and orthogonalization with Z-axis fixed in blue.

other methods do not directly consider diffusing orientations but they first consider smoothness of the vector diffusion and only later enforce orientation constraint. Although the SVD orthonormalized vector diffusion approach provides a very good approximation for the Spot point cloud, the difference might be higher for more complex shapes.

B.2 Supplementary Results

B.2.1 Task-level Quantitative Evaluations

We selected robotic peeling as a representative benchmark to provide a task-level quantitative evaluation of our method. Because our core contribution is a shape-invariant geometric representation, we focus on demonstrating its generalization across varying geometries rather than

Figure B.3: **Real-world peeling results**. We show peels and the final state after peeling. Dashed black lines indicate broken peels during the peeling (counted as failure) and dashed blue lines indicate broken peels during the transfer after the peeling (counted as success, see Movie S1). Green boundaries show the whole area and red boundaries highlight unpeeled regions used to compute the peeled area.

absolute peeling performance, which depends heavily on specific objects peeled, hardware setup and low-level controllers. Therefore, we tested our approach on a diverse set of objects with complex, bent, and irregular shapes.

To enable a direct comparison with recent imitation learning approaches, we selected cucumbers and zucchinis to match the experimental setups of Reactive Diffusion Policy (RDP) [103] and ForceMimic [62], respectively. To further challenge our approach, we evaluated it on three additional objects with irregular and bent shapes: a carrot (hard skin), a sweet potato (hard skin), and a pear (soft skin), contrasting the medium-skinned zucchini and cucumber. For each object, we peeled the entire skin by manually re-orienting the object and annotating the endpoints. For performance metrics, we report (i) continuous peeling length [103, 62] and (ii) peeled skin area [28, 107] in Table B.2, utilizing the performance values reported in their respective papers. Following [62], continuous peeling length is calculated as the ratio of peels that are continuous along the object's length to the total number of peels. As proposed by [107], the peeled area is computed by evaluating the ratio of unpeeled area to the total area in pixel space, utilizing two images that capture the unpeeled regions. We provide all annotated peels and peeled areas used for metric computation in Figure B.3. Experimental data, recorded as bag files, are available in our repository. We provided the experiment videos as a supplementary video (Movie S1). Because our hardware setup was not specifically designed for peeling, a human operator manually assisted the fixture to prevent the object from pivoting and to keep the cables from tangling during the peeling process.

Our approach achieves comparable performance to state-of-the-art imitation learning meth-

Object	Method	Continuous Peeling Length	Peeling Area
Cucumber	DP [103]	0.56	-
	RDP [103]	0.99	-
	MORPHeus [107]	-	0.93
	Bimanual [28]	-	0.94
	DiffOF	1.00	0.97
Zucchini	DP [62]	0.55	-
	ForceMimic [62]	0.85	-
	MORPHeus [107]	-	0.84
	DiffOF	1.00	0.89
Carrot	MORPHeus [107]	-	0.93
	DiffOF	0.86	0.95
Pear	**DiffOF**	0.79	0.96
Sweet Potato	**DiffOF**	0.95	0.95

Table B.2: **Task-level evaluation for peeling task.** We abbreviated diffused orientation fields as DiffOF. We report the continuous peeling length and peeled area achieved by our approach. For comparison, we include previously reported metrics from state-of-the-art imitation learning and planning methods evaluated on identical object categories.

ods on zucchini and cucumber without requiring any demonstrations or training (see Table B.2). These prior works were trained and evaluated on only a single object type featuring simple, straight, cylindrical shapes (cucumbers in [103] and zucchinis in [62]). Consequently, their performance on more complex geometries, or even on bent and irregular cucumbers and zucchinis, is unknown. RDP is trained with 60 demonstrations on cucumbers, whereas ForceMimic utilizes 438 demonstrations on zucchinis and both of these methods rely on specialized interfaces to collect demonstrations. Scaling such demonstration-based methods to the large variety of objects tested in our setup would quickly become unmanageable. By contrast, we evaluate our method across diverse object geometries, including bent and irregular shapes, highlighting the principal advantage of our shape-invariant representation. We show our method maintains consistent peeled area performance across all objects (see Table B.2) and produces geometrically correct peeling motions, as demonstrated in our experiment videos (see Movie S1).

However, as expected, the continuous peeling length performance decreases on the harder and softer skin objects: carrot, sweet potato, and pear. This degradation occurs for two reasons: first, unlike zucchinis and cucumber generating long, continuous peels on these challenging objects is difficult even by a human; second, our compliant controller parameters were not explicitly tuned for these varying material properties, resulting in insufficient compliance for hard skins and excessive compliance for soft skins. A promising future direction is to integrate imitation learning approaches [103, 62] with our representation. This would allow the system to learn compliance parameters within the local frames, thereby enabling better geometric and force-interaction transfer across diverse objects.

As can be seen in Table B.2, our method achieves comparable peeled area performance to planning approaches [28, 107, 22] without assuming the objects possess axis-symmetry (such as cylindrical or spherical shapes). In fact, with the exception of the cucumber, all objects tested in our real-world experiments deviate considerably from axis-symmetry. To highlight the advantages of our shape-invariant representation over the axis-symmetry assumption, we designed a kinematic peeling benchmark to measure the resulting peeled areas in a controlled and repeatable setup. For this evaluation, we generated a capsule object and gradually bent it perpendicular to its centerline to break axis-symmetry. We then implemented the baseline axis-symmetric peeling strategy [28, 22], which computes a rotation axis connecting the two endpoints, generates rotated planes around this axis, and calculates their intersections with the object's surface. Following [28], we only consider the intersection curve on one side of the axis, as the opposite side is typically unreachable by the robot. When axis-symmetry breaks, restricting intersections to one side is insufficient to prevent unreachable paths; however, to ensure a highly competitive baseline, we nevertheless counted all such generated points as properly peeled. Figure B.4 visualizes the bent capsules and plots the peeled and over-peeled areas versus the number of peels.

On a perfectly straight capsule, our representation matches the optimal performance of the axis-symmetric baseline (Fig. B.4A). However, as increased bending violates the axis-symmetry assumption, the baseline's performance drops substantially. Conversely, our shape-invariant representation remains robust to curvature variations, maintaining optimal performance throughout without making assumptions on the object shape (Fig. B.4B-C). Although our representation yields superior trajectories, the peeling pipelines proposed in [28, 107, 22] offer more complete peeling systems with object reconfiguration and fixturing mechanisms, which are beyond the scope of our work. That being said, integrating our shape-invariant representation into these existing peeling pipelines presents a promising direction for developing a fully comprehensive peeling system.

Finally, our representation can easily express more complex, longer-horizon sequences composed of multiple connected peeling motions in alternating directions (see Movie S1). For a given object configuration, the robot can perform several continuous peels to comprehensively clear the reachable and observable surface before requiring any re-configuration. As we demonstrated, executing three consecutive peels reduces the required number of object reconfigurations by a factor of three. Furthermore, the ability to peel bi-directionally eliminates the need to reset the tool position between strokes, reducing the total end-effector displacement by at least half.

B.2.2 Computational Performance

We evaluated the computational performance of our method on point clouds with varying numbers of vertices, using a laptop equipped with an Apple M1 Pro chip and 32 GB of RAM. The results are presented in Table B.3.

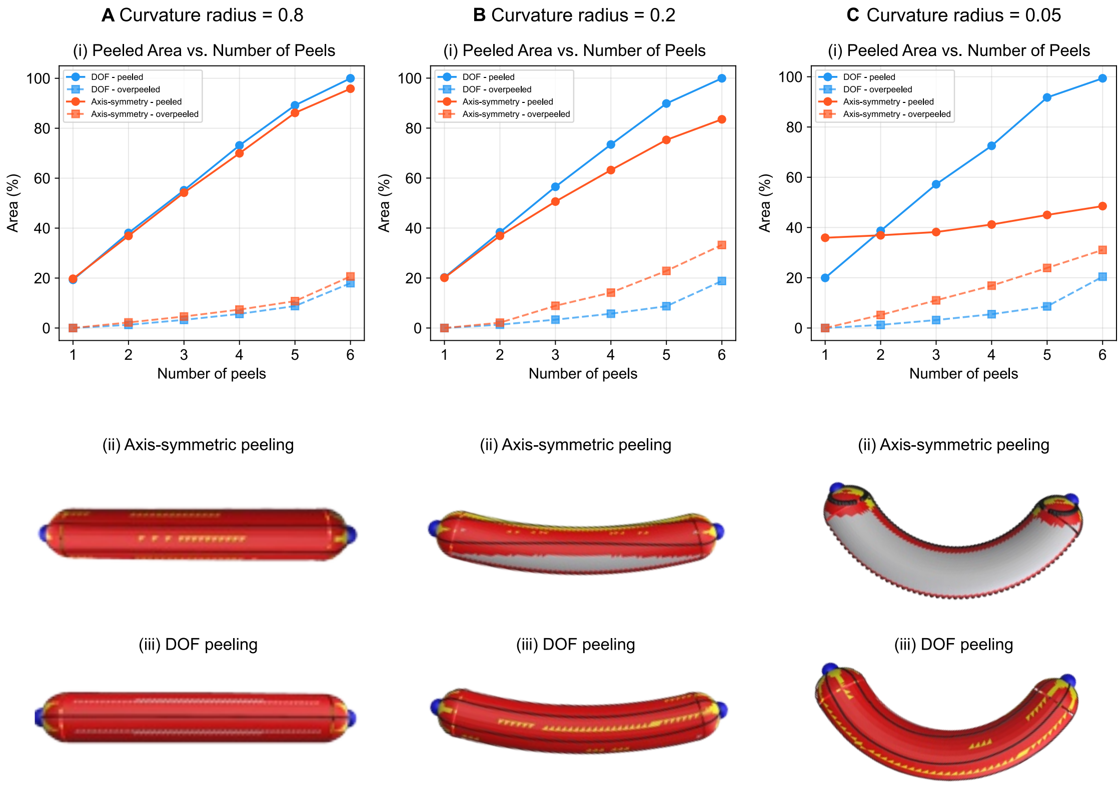


Figure B.4: **Effect of object curvature on peeling performance.**.(**A–C**) Progression from left to right demonstrates the capsule object with increasing bending (decreasing radius of curvature). (i) Line plots evaluate orientation fields versus axis-symmetric peeling by comparing peeled and overpeeled areas. (ii–iii) Visualizations illustrate the bent object and corresponding peeling areas for tested methods. Peeled, overpeeled, and unpeeled regions are highlighted in red, yellow, and gray, respectively.

Object	**Points**	**Laplacian**	**Factorization**	**Surface Diffusion**	**WoS**
Bowl	3,000	58 ms	10 ms	0.3 ms	4 ms
Spot	6,000	133 ms	27 ms	0.6 ms	7 ms
Bunny	12,000	244 ms	44 ms	1.1 ms	26 ms

Table B.3: **Average computation times for the main components of our method.** For the Walk on Spheres (WoS), we used 256 samples per query and considered a sample to have converged when it was within twice the average neighbor distance from the boundary.

As shown in Table B.3, preprocessing time is primarily dominated by the computation of the discrete Laplacian. Even in complex scenes (for example, those involving four distinct objects with 12k vertices each), the total preprocessing time remains under one second. Crucially, the discrete Laplacian is invariant under isometric transformations. Isometric transformations include rigid body motions or bending without stretching, which account for most of the transformations encountered in robotic manipulation. This makes diffused orientation fields particularly well-suited to such settings, as the Laplacian does not need to be recomputed when the geometry undergoes these typical changes.

At runtime, orientation field's value representing the local reference frame at the robot's position is computed using the Walk on Spheres (WoS) method. WoS is a Monte Carlo technique similar to ray casting, well-suited for massive parallelization and hardware acceleration. Despite being implemented in pure Python on the CPU, without adaptive step sizing, importance sampling, or boundary value caching, still our method enables real-time operation. Importantly, since WoS is inherently parallelizable, the runtime would remain constant with respect to the number of query points when executed on a GPU, making diffused orientation field evaluation highly scalable.

B.2.3 Comparison with Alternative Baselines

We compared our method to three alternative baselines for computing local reference frames in terms of smoothness. We selected these baselines to isolate different components of our method, allowing us to evaluate the importance of each component. We provide a detailed description of each baseline and the results of the comparisons below.

Nearest Frame Baseline

Nearest frame baseline keeps the diffused surface orientation field but replaces the workspace diffusion with the nearest frame projection. Specifically, for each point in the workspace, we find the nearest point on the surface and use the local frame at that point as the local frame at the query point. We then compared our method (workspace diffusion) to the nearest frame baseline. We quantified smoothness using the average angular difference between neighboring local x directions. We measured $\approx$ 1.5x higher average angular difference for the nearest frame baseline compared to our method, indicating that workspace diffusion produces a smoother field. Note that this is an expected result since we can think of workspace diffusion as a smoothed projection to the boundary. We provided the results of this comparison in Fig. B.5. As we show Fig. B.5, iii, the nearest frame baseline produces discontinuities in the local frames, resulting in undesired behavior for the slicing task. Also, in other downstream robotics tasks, the discontinuities in the local frames would lead to force spikes and jerky motion.

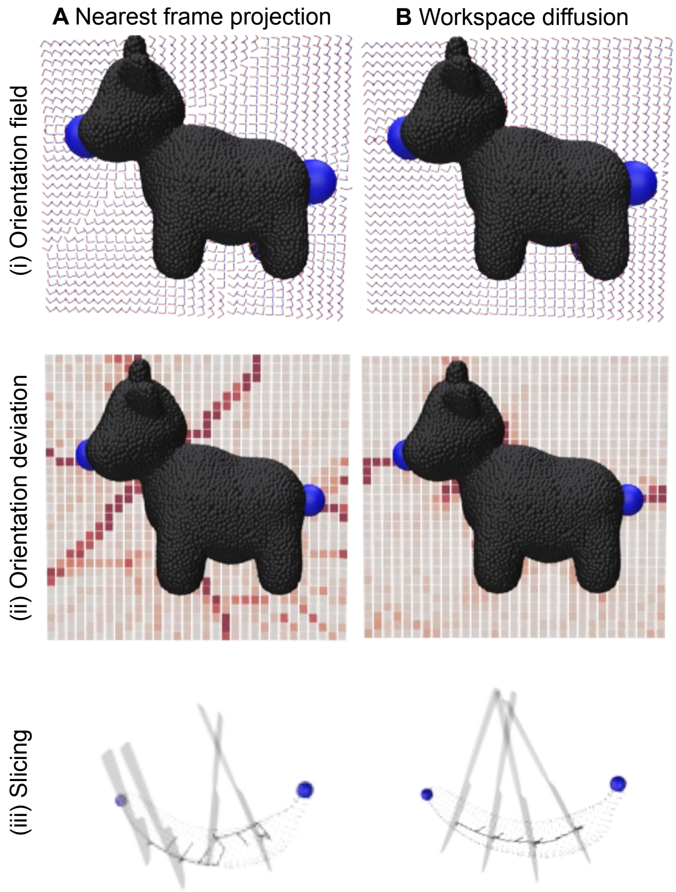


Figure B.5: **Comparison between nearest frame baseline and workspace diffusion**. (**A**) Nearest frame projection results in discontinuity in the orientation field since the nearby workspace points are mapped to distant points on the point cloud. (**B**) Workspace diffusion (ours) resulting in smooth orientation field. (i) A slice of the orientation field on Spot's symmetry plane. (ii) Deviation of the orientation in the local neighborhood shown using a heat map. (iii) Slicing task using nearest frame baseline resulting in undesired discontinuity due to the non-smooth orientation field.

Vector Projection Baseline

Vector projection baseline complements the nearest frame baseline as it provides a comparison for surface diffusion component of our approach. For computing the x-direction at a query point, we considered the weighted sum of the vectors connecting the query point to the keypoints. Next, we projected this vector to the tangent plane at the query point. One can compute the tangent plane at any point in the workspace, which is given by the gradient of the signed distance field to the surface.

We compared this baseline to our approach by using a number of spurious sources/sinks (undesired singularities) and visualized the results in Fig. B.6. We marked the spurious sources/sinks by using average angular difference between neighboring local x directions. As can be seen in Fig. B.6, i-ii, the vector projection baseline produced spurious sources/sinks (9 in total for the Spot point cloud) in the vector field, whereas our method is free of unaccounted singularities other than the ones resulting from keypoints. Note that, although it seems in Fig. B.6B, ii that our method produces a singularity in the horns of the Spot, this is an artifact of our angular deviation computation (error introduced by projecting neighboring vectors to the tangent space). We show this by visualizing the vector field in Fig. B.6C. Due to spurious sources/sinks and non-smoothness, the vector projection baseline is not suitable for downstream robotics tasks. Moreover, we show that even if there are no spurious sources/sinks (since the vector projection baseline ignores the geodesic paths on the surface during the diffusion), the approach fails to capture the global symmetry of the object, leading to undesired behavior for the slicing task (Fig. B.6, iii).

Euclidean Diffusion Baseline

As the third baseline, we considered Euclidean diffusion using keypoints (ignoring the surface geometry during the diffusion), computing its gradient field to get x-direction and projecting the resulting vectors to the tangent space. For computing the tangent space we used smooth extension of surface normals to the workspace using WoS. This baseline addresses smoothness issues related to the first two baselines. However, it ignores the geometry of the surface during the diffusion of the x-direction and extrinsically correct it afterwards. We used a T-shaped object for this comparison since it has a concave region which shows the issue created by ignoring the surface geometry for diffusing x-directions. The Euclidean diffusion baseline produces x-directions that cut across the concavity of the T-shape (Fig. B.7A). This is an undesired behavior since it would lead to undesired motion for downstream robotics tasks. On the other hand, our method respects the geometry of the surface during diffusion and produces x-directions that follow the surface layout (Fig. B.7B).

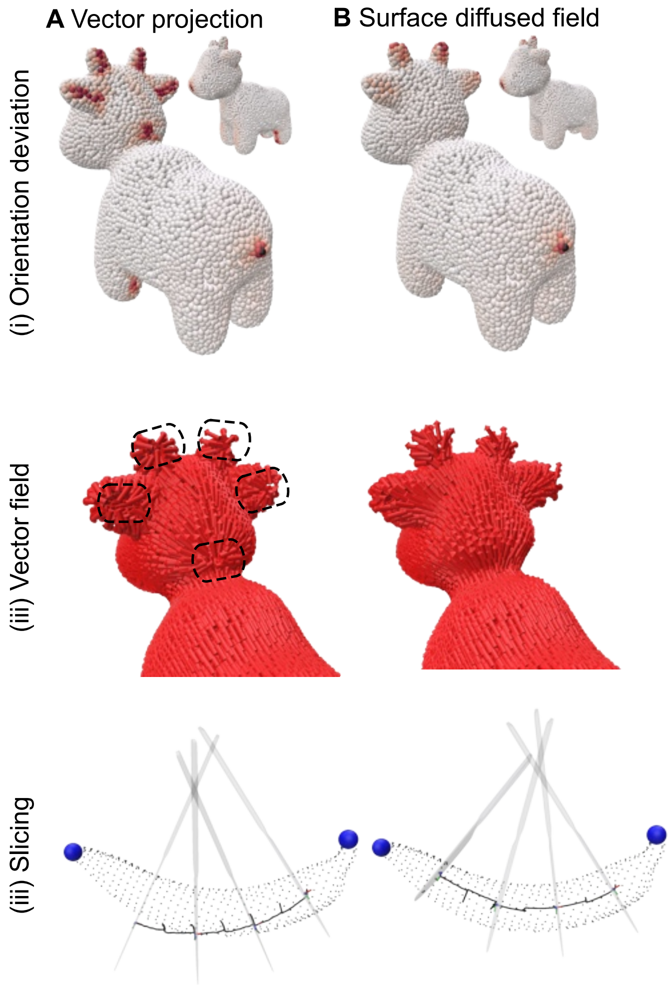


Figure B.6: **Comparison between vector projection baseline and surface diffusion**. (**A**) Vector projection baseline. Vector projection results in a non-smooth vector field on the tangent planes leading to spurious sources and sinks. (**B**) Surface diffusion (ours) producing a smooth vector field on the tangent planes free of sources and sinks except the ones at keypoints. (i) Orientation deviation on the tangent plane shown using a heat map. (ii) Deviation of the orientation in the local neighborhood. Higher red intensity show higher deviation. (iii) Slicing task comparison. We visualize the slicing trajectory with the black curve and visualize the tool pose in 4 equally-spaced time intervals.

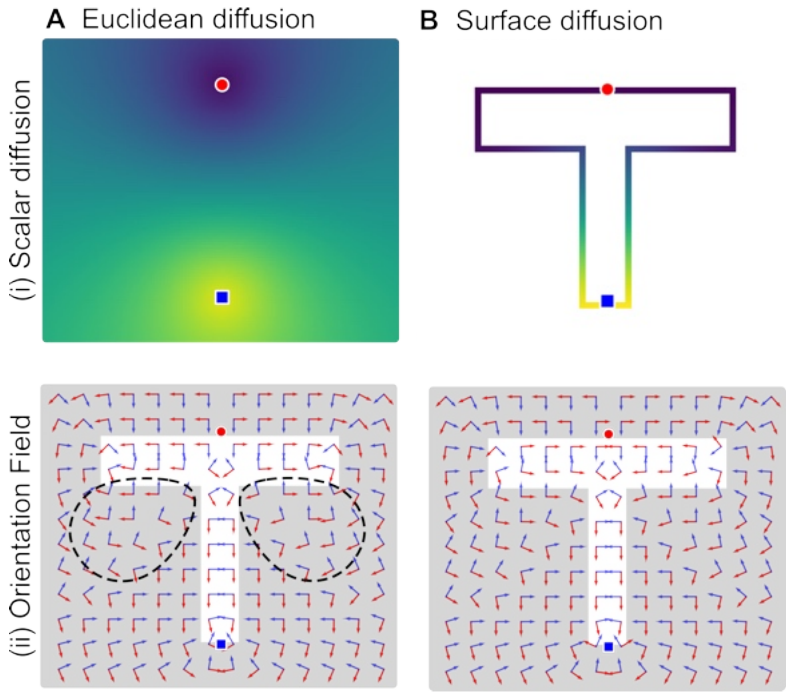


Figure B.7: **Comparison between Euclidean diffusion baseline and surface conditioned workspace diffusion**. Red point designates the source whereas the blue square is the sink. (**A**) Euclidean diffusion baseline, ignoring the surface geometry during the diffusion and using projection of the gradient vectors to the tangent space. (**B**) Diffused orientation field (ours). (i) Results of scalar Euclidean and surface diffusion. (ii) Orientation field comparison. Regions where the Euclidean diffusion baseline cuts across the concavity of the T-shape are highlighted with dashed lines.

Table B.4: **Hyperparameters.**

Group	**Parameter**	**Value**
	Diffusion Time (τ)	1000 (10 only for coverage)
	Voxel Downsampling	3 mm
Admittance Controller	Position Stiffness (N/m)	[400,200,200]
	Rotational Stiffness (Nm/rad)	[10,10,20]

B.2.4 Robustness to Clutter and Occlusions

Our method is robust to clutter: objects that are close in Euclidean distance but far in geodesic distance on the target surface have limited influence on the diffusion. Since the diffusion follows geodesics, distractor surfaces do not alter geodesic paths along the object. We show this in a cluttered scene in Fig. B.8A: despite nearby distractors, the slicing trajectory is unaffected. A second case is occlusion, where the object of interest is partially hidden. If the occlusion does not split the object into disconnected regions, we do not observe problems (Fig. B.8B). Although occluders perturb local shape, they do not break the object's global symmetry, so the trajectory remains stable. Naturally, if the occluder dominates the view, performance degrades. In such cases, the occluder should first be removed or the camera viewpoint changed. A problematic case arises when an occluder breaks the observed geometry into disconnected regions (Fig. B.8C). We placed a cable across the object so that it occluded the midsection of a banana, producing two disconnected components. With zero-Neumann boundary conditions (which act as diffusion insulators), the diffusion "followed" the cable and led to failure. To address this, a lightweight instance-segmentation step could be introduced to identify whether disconnected components belong to the same object. This approach would allow the system to differentiate between boundary types, applying the zero-Neumann condition exclusively to external boundaries rather than internal ones. This both preserves diffusion across the object and insulates against true external occluders. We made a successful proof-of-concept test for this approach using Grounded SAM [77] for segmenting the objects and figuring out the disconnected regions belong to the same object, as shown in Fig. B.8D.

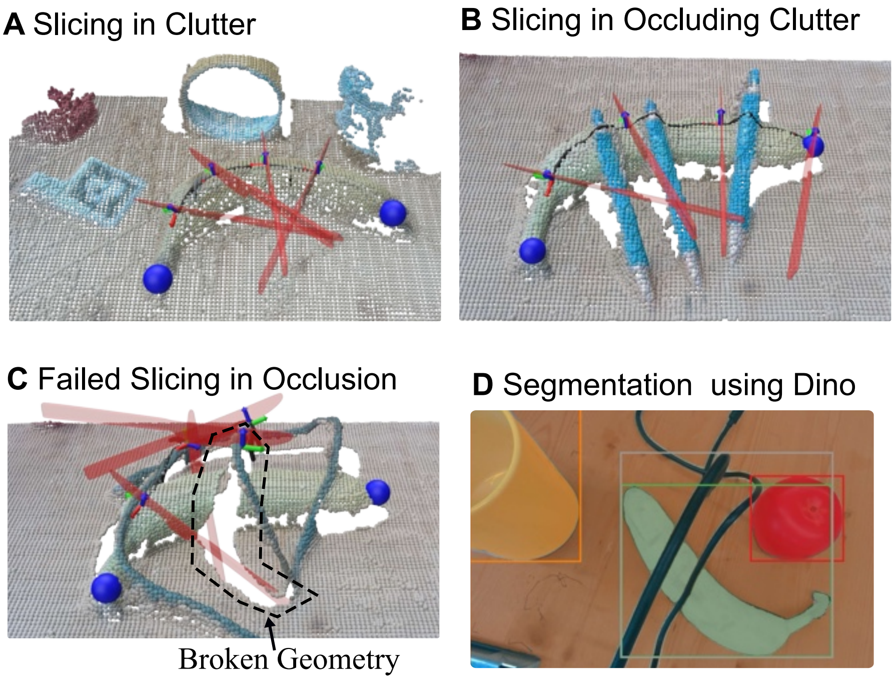


Figure B.8: **Expriments on cluttered scenes.** (**A**) Slicing in the presence of distractor objects. (**B**) Slicing in the presence of an occluder that does not break the object into disconnected regions. The occluder perturbs local shape but does not break the object's global symmetry, so the trajectory remains stable. (**C**) Slicing in the presence of an occluder that breaks the object into disconnected regions. We placed a cable across the object so that it occluded the midsection of a banana, producing two disconnected components. With zero-Neumann boundary conditions (which act as diffusion insulators), the diffusion "followed" the cable and led to failure. (**D**) Proof-of-concept experiment to address the failure cases in (**C**) using Grounded SAM [77]. We use object segmentation for detecting that the disconnected regions belong to the same object.