Tactile Ergodic Coverage on Curved Surfaces

Cem Bilaloglu*, Tobias Löw*, and Sylvain Calinon

***Abstract*—In this article, we present a feedback control method for tactile coverage tasks, such as cleaning or surface inspection. These tasks are challenging to plan due to complex continuous physical interactions. In these tasks, the coverage target and progress can be easily measured using a camera and encoded in a point cloud. We propose an ergodic coverage method that operates directly on point clouds, guiding the robot to spend more time on regions requiring more coverage. For robot control and contact behavior, we use geometric algebra to formulate a task-space impedance controller that tracks a line while simultaneously exerting a desired force along that line. We evaluate the performance of our method in kinematic simulations and demonstrate its applicability in real-world experiments on kitchenware. Our source codes, experimental data, and videos are available as open access at https://sites.google.com/view/tactile-ergodic-control/.**

***Index Terms*—Tactile Robotics, Ergodic Coverage, Geometric Algebra**

I. INTRODUCTION

The longterm vision of robotics is to assist humans with daily tasks. Especially the success of robot vacuum cleaners and lawnmowers as consumer products demonstrates the potential of robot assistance with the most common household chores [1]. These tasks involve the coverage of a region in a repetitive and exhaustive fashion. Currently, the application of these robots is limited to relatively large and planar surfaces and even their use on slopes poses a challenge [2], [3]. Other daily tasks, such as washing the dishes or grocery items present an even more challenging problem due to the complex physical interactions with intricate and curved surfaces. Similarly, there are numerous coverage tasks on curved surfaces that have industrial or medical applications. In industrial settings, these problems manifest in two forms: surface operations that involve the removal of material, such as sanding [4], polishing [5], [6] or deburring [7] and surface inspection tasks that leverage contacts [8]. In medical settings, similar applications range from mechanical palpation [9], [10] and ultrasound imaging [11], [12] to massaging [13], [14] and bed bathing [15], [16]. Last but not least, datasets combining tactile properties of objects with their shape and visual appearance are extremely scarce and expensive to collect, since they are based on teleoperation [17]. Hence, tactile coverage is of paramount importance for automating the collection of the tactile datasets complementing the visual ones. The problem definitions of this diverse range of settings and applications can be broken down to two simple requirements: (i) tactile interactions with a possibly non-planar surface and (ii) a continuous trajectory of contact points covering a region of interest on the surface. Accordingly, the overarching problem that is tackled in this article is posed as tactile coverage on curved surfaces. Tactile interaction tasks, by definition, involve multiple contact interactions with the environment, making these systems notoriously difficult to control [18]. While these tasks are easily solved by humans, they are extremely challenging for robots. For instance, when cleaning an object, coverage depends on recognizing the dirt, the object's material, and their interaction to determine the required contact force to remove it. Consequently, the success of coverage depends on unknown or difficult-to-measure parameters, making it hard to model all interactions, and motion planning without a proper model is prone to fail. By analyzing previous research [19] and how humans address these challenges, we argue that, instead of planning humans solve the easier closed-loop control problem by leveraging visual and tactile feedback for online adaptation. Similar to humans, robots can also measure progress in tactile coverage tasks using vision. Determining which regions of the surface have already been covered and which have not then becomes an image segmentation problem, that was addressed by leveraging various model [20], [21] or learning-based

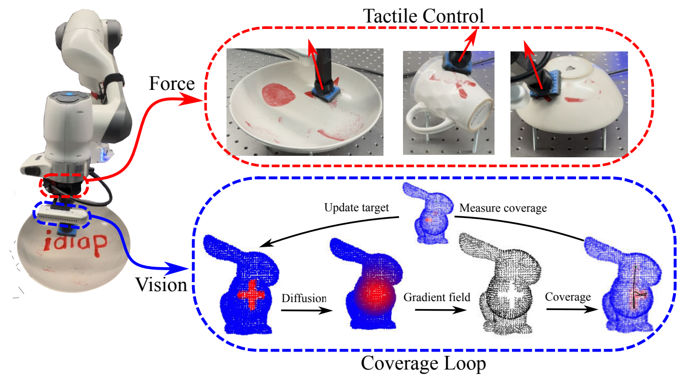

Fig. 1: Overview of our feedback control method for tactile coverage. *Left:* We measure the surface and the red target using the camera and encode them in a point cloud. *Bottom-right:* We diffuse the target and use its gradient field to guide the coverage. Then, we close the loop by measuring the actual 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.

This work was supported by the State Secretariat for Education, Research and Innovation in Switzerland for participation in the European Commission's Horizon Europe Program through the INTELLIMAN project (https://intelliman-project.eu/, HORIZON-CL4-Digital-Emerging Grant 101070136) and the SESTOSENSO project (http://sestosenso.eu/, HORIZON-CL4-Digital-Emerging Grant 101070310).

*Equal contribution. The authors are with the Idiap Research Institute, Martigny, Switzerland and with the Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland. cem.bilaloglu@idiap.ch; tobias.loew@idiap.ch; sylvain.calinon@idiap.ch

algorithms [16], [22]. Still, the question remains on how to control the robot to cover these target regions on the surface.

Existing research on coverage has primarily focused on coverage path planning, which involves optimizing a path to ensure that a specified region of interest is covered within a set time frame. Traditionally, the underlying assumption is that visiting each point in the region of interest only once is sufficient for full coverage. An assumption that is reasonable for robust interactions but not for many tactile tasks. Because many tactile interactions are too complex to model deterministically, hence the full coverage cannot be guaranteed after a single visit. Instead, for a cleaning task, a relatively dirty region requires more visits compared to a less dirty region. Similarly, in a surface inspection task, a region where we require a lower variance requires more visits to compensate for the uncertainty of the sensor. Furthermore, the robot is expected to keep in contact with the surface while moving, which significantly increases the cost of movement, since it depends on the geodesic distance on the surface and not on the Euclidean distance. Therefore, naive sampling strategies that do not consider the cost or constraints of the movement and/or the surface geometry are not suited for tactile coverage tasks. In contrast, ergodic coverage [23] controls the trajectories of *dynamical systems* for ergodicity, correlating the average time spent in a region to the target spatial distribution. Therefore, ergodic coverage can incorporate the motion model as the system dynamics and control the coverage trajectories by directly using the spatial distribution measured by the vision system.

Considering these challenges, we present a closed-loop tactile ergodic control method that operates on point clouds for tactile coverage tasks. Using point clouds not only allows us to acquire the target object and spatial distribution at runtime using vision, but to also measure the coverage progress and to compensate for unmodeled dynamics of the tactile coverage tasks. Our method then constrains the ergodic control problem to arbitrary surfaces to cover a target spatial distribution on the surface. We propagate the coverage information by solving the diffusion equation on point clouds. We solve the diffusion in real-time by exploiting the surface intrinsic basis functions called Laplacian eigenfunctions, generalizing the Fourier series to manifolds (i.e., curved spaces). In order to exert a desired force on the surface while moving, we then formulate a geometric task-space impedance controller using geometric algebra. This controller utilizes the surface information to track a line target that is orthogonal to the surface, while simultaneously exerting the desired force in the direction of that line. Notably, the geometric formulation ensures that these two objectives do not compete with each other and can therefore be included in the same control loop, without the need for complex parameter tuning. In summary, our proposed closed-loop tactile ergodic control method has the following contributions:

- formulating the tactile coverage as closed-loop ergodic control problem on curved surfaces
- closing the coverage loop by solving ergodic control problem on point clouds using diffusion
- achieving real-time frequencies by computing the diffusion using Laplacian eigenfunctions
- contact line and force tracking without conflicting objectives

The rest of the article is organized as follows. Section II describes work related to our method. Section III presents the mathematical background. Section IV presents our method. In Section V, we demonstrate the effectiveness of our method in simulated and real-work experiments. Finally, we discuss our results in Section VI.

II. Related Work

In this article we address the problem of tactile coverage on curved surfaces. There are various approaches that consider the coverage problem from the planning perspective and are generally known as coverage path planning (CPP) [24]–[26]. Although, these methods can consider different types of boundaries for planar regions [27]–[29], their extension to curved surfaces imposes limiting assumptions, such as projectively planar [30] or pseudo-extruded surfaces [31]. Additionally, CPP methods assume the coverage target is uniformly distributed in space. Extension of the CPP methods that consider the spatial correlation of the information are known as informative path planning [32]. Most IPP and CPP approaches solve a variant of the NP-hard traveling salesman problem [33], limiting the scalability with the complexity of the domain. Therefore existing methods are either open-loop [34] or pose limiting assumptions on the domain for online planning updates [32] such as convexity.

In this article we focus on tactile coverage scenarios, in which visiting a region once does not guarantee it is fully covered. Thus, it is hard to predict how many times the robot should pass over a certain spot. Consequently, we cannot define a time horizon for an optimization, since the quality of the result would be greatly affected by that hard-to-make choice. In this work, we address this issue by using an approach called ergodic control. In this context, the term ergodic describes a dynamical system for which the time averages of functions along its trajectories are equal to their spatial averages [35]. The important consequence of this is that it allows us to use arbitrary spatial target distributions without having to define a time horizon, since, by construction, regions with higher spatial probabilities will be visited more frequently. Recent findings have demonstrated that ergodicity is not merely a heuristic [36]; it is the optimal method for collecting independent and identically distributed data while accounting for system dynamics. Seminal work on ergodic control, presented the spectral multiscale coverage (SMC) algorithm [23] that provides a feedback control law based on the Fourier decomposition of the target distribution and robot trajectories. Here, multiscale coverage refers to the prioritization of the low-frequency components over high-frequency ones, which intuitively corresponds to first using large spatial motions in the coverage before getting into the details. Since this behavior is obtained through a myopic feedback controller, unlike an offline planner, the ergodic controller would not fail if the motion is obstructed [37].

Formulating the SMC objective as trajectory optimization then allowed the explicit consideration of obstacle avoidance [38] and other additional objectives such as time-optimality [39] and energy-awareness [40]. These formulations are limited to rectangular domains on the Euclidean space, since they are based on the Fourier decomposition. Later, SMC was extended to homogeneous Riemannian manifolds using the Laplacian eigenfunctions [41]. Nevertheless, this work was limited to highly structured manifolds such as sphere and torus, due to the requirement of the analytical expressions for the Laplacian eigenfunctions. More recently, the *kernel ergodic metric* was proposed as an alternative to SMC's ergodic metric to increase the computational efficiency and for extension to Lie groups [42]. Another alternative to SMC is the heat equation-driven area coverage (HEDAC) [43]. HEDAC uses the diffusion equation to propagate information regarding the uncovered regions to agents across the domain. Like SMC, the original HEDAC implementation was limited to rectangular domains and lacked collision avoidance. It has since been extended to planar meshes with obstacles [44], maze exploration [45], and CPP on non-planar meshes [46]. However, the implementation on curved surfaces is limited to offline planning on meshes and demands heavy pre-processing in terms of time and computation.

Our method is inspired by both HEDAC and SMC. We use the diffusion equation for the information propagation over the surface and we use the Laplacian eigenfunctions, which generalize the Fourier series to manifolds for efficient computation. The diffusion equation is a canonical second-order partial differential equation (PDE), which propagates information on a domain by considering its geometry, while agnostic to underlying representation and discretization [47]. Therefore, the diffusion equation and its spectrum are used in various geometry processing tasks on meshes and point clouds, ranging from geodesic computation [48] to learning on surfaces [47]. Although various approaches exists for solving the diffusion, the most common one is to use the *Laplace-Beltrami* operator, extending the Laplacian from Euclidean space to curved spaces. For a given point cloud, there are various methods for computing the Laplace-Beltrami operator [49]–[51]. Sharp *et al.* provide a robust and efficient implementation [52], even in the case of partial and noisy point clouds. We use this approach for solving the ergodic control problem on arbitrary point clouds for tactile coverage.

Closely related to coverage is the problem of exploration, which involves scenarios where the environment is initially unknown and robots collect information about the environment using onboard sensors [53], [54]. Tactile ergodic exploration was employed for non-parametric shape estimation [55] and whole-body coverage using all the link surfaces [56]. However, these works were limited to rectangular domains in the Euclidean space. Tactile exploration is also needed for gathering the information on surfaces that can only be acquired through contact [57], unlike surface reconstruction or localization. A prime example of this is non-invasive probing (palpation) of tissue stiffness, which can aid in disease diagnosis or surgery by providing additional information about anatomical features. For that purpose, Gaussian processes (GP) were employed for discrete [58] or continuous [59] probing to map tissue stiffness. While GP-based formulations provide guidance on where to sample, they are unaware of the robot's dynamics. This was addressed by using trajectory optimization to actively search for tissue abnormalities [60]. However, the critical aspect of tactile interactions is that they not only depend on the contact position but also on other contact conditions such as relative velocity and contact pressure [61]. Therefore, there are also methods modeling the force [62] and more complex interactions between robotic tools and surfaces [63].

The complexity of the problem increases further if we consider scenarios with a robot physically interacting with the environment. For example, in tasks like surface finishing (e.g., polishing, sanding, grinding), the surface itself undergoes changes, as material is removed [64]. Similarly, in cleaning tasks, the robot's actions affect the distribution of dirt on the surface [65]. To avoid complex modeling, there are approaches either relying on reinforcement learning [66] or deep learning [67]. Learning from demonstration has also been used by also leveraging ergodicity for table cleaning [68], where different motion trajectories can achieve the same task as long as they result in the same ergodic coverage. In a very similar setting to ours, a manipulator was used to clean the stains on a curved surface by performing multiple passes [20]. However, this work used a sampling-based planner, which required to predefine the maximum number of cleaning passes. In contrast, we relate the target distribution (e.g., stain) directly to feedback control through ergodicity without requiring any task-specific assumptions.

III. BACKGROUND

A. Ergodic Control using Diffusion

The ergodic control objective correlates the time that a coverage agent spends in a region to the probability density specified in that region. The HEDAC method [43] encodes the coverage objective using a virtual source term

$$s(\boldsymbol{x},t) = \max\left(p(\boldsymbol{x}) - c(\boldsymbol{x},t), 0\right)^2, \tag{1}$$

where $p(\boldsymbol{x})$ is the probability distribution corresponding to the coverage target and $c(\boldsymbol{x},t)$ is the normalized coverage of the N virtual coverage agents over the domain Ω

$$c(\boldsymbol{x},t) = \frac{\tilde{c}(\boldsymbol{x},t)}{\int_\Omega \tilde{c}(\boldsymbol{x},t) d\boldsymbol{x}}. \tag{2}$$

A single agent's coverage is the convolution of its footprint $\varphi(\boldsymbol{r})$ with its trajectory $\boldsymbol{x}_i(\tau)$. Then, the total coverage becomes the time-averaged sum of these convolutions

$$\tilde{c}(\boldsymbol{x},t) = \frac{1}{Nt}\sum_{i=1}^{N}\int_0^t \varphi\left(\boldsymbol{x} - \boldsymbol{x}_i(\tau)\right) d\tau. \tag{3}$$

HEDAC diffuses the source to the whole domain by computing the resulting potential field $u(\boldsymbol{x},t)$ using the stationary ($\dot{u} = (\boldsymbol{x},t)0$) diffusion (heat) equation with the diffusion coefficient $\alpha > 0$

$$\alpha \Delta u(\boldsymbol{x},t) - u(\boldsymbol{x},t) + s(\boldsymbol{x},t) = 0. \tag{4}$$

In order to have a unique solution, we need to prescribe the initial and boundary conditions

$$u(\boldsymbol{x}, 0) = p(\boldsymbol{x}) \quad \text{and} \quad \frac{\partial}{\partial \boldsymbol{n}} u(\boldsymbol{x}, t) = 0, \quad \text{on } \partial\Omega. \tag{5}$$

In the diffusion equation, (4) Δ denotes the second-order differential operator. In Euclidean spaces (isotropic) the Laplacian is the sum of the second partial derivatives

$$\Delta f = \sum_{i=1}^{n} \frac{\partial^2 f}{\partial \boldsymbol{x}_i^2} \quad \text{for} \quad \boldsymbol{x}_i \in \mathbb{R}^n. \tag{6}$$

To guide the i-th coverage agent, HEDAC uses the smooth gradient field of the diffused potential $u(\boldsymbol{x}, t)$ and by simulating first-order dynamics [69]

$$\dot{\boldsymbol{x}}_i = \nabla u(\boldsymbol{x}_i, t). \tag{7}$$

B. *Conformal Geometric Algebra*

Here, we introduce conformal geometric algebra (CGA) with a focus on the mathematical background necessary to understand the methods used in this article. We will use the following notation throughout the paper: x to denote scalars, $\boldsymbol{x}$ for vectors, $\boldsymbol{X}$ for matrices, X for multivectors and $\boldsymbol{\mathcal{X}}$ for matrices of multivectors.

The inherent algebraic product of geometric algebra is called the geometric product

$$\boldsymbol{a}\boldsymbol{b} = \boldsymbol{a} \cdot \boldsymbol{b} + \boldsymbol{a} \wedge \boldsymbol{b}, \tag{8}$$

which (for vectors) is the sum of an inner $\cdot$ and an outer $\wedge$ product. The inner product is the metric product and therefore depends on the metric of the underlying vector space over which the geometric algebra is built. The underlying vector space of CGA is $\mathbb{R}_{4,1}$, which means there are four basis vectors squaring to 1 and one to -1. The outer product, on the other hand, is a spanning operation that effectively makes subspaces of the vector space elements of computation. These subspaces are called blades. In the case of CGA, there are 32 basis blades of grades 0 to 5. The term grade refers to the number of basis vectors in a blade that are factorizable under the outer product. Vectors, consequently, are of grade 1 and the outer product of two independent vectors, called bivectors, are of grade 2. A general element of geometric algebra is called a multivector.

In practice, CGA actually applies a change of basis by introducing the two null vectors $\boldsymbol{e}_0$ and $\boldsymbol{e}_\infty$, which can be thought of as a point at the origin and at infinity, respectively. Since the Euclidean space is embedded in CGA, we can embed Euclidean points $\boldsymbol{x}$ to conformal points P via the conformal embedding

$$P = \mathcal{C}(\boldsymbol{x}) = \boldsymbol{e}_0 + \boldsymbol{x} + \frac{1}{2}\boldsymbol{x}^2 \boldsymbol{e}_\infty. \tag{9}$$

In general, geometric primitives in geometric algebra are defined as nullspaces of either the inner or the outer product, which are dual to each other. The outer product nullspace (OPNS) is defined as

$$\mathbb{NO}_G(X) = \left\{ \boldsymbol{x} \in \mathbb{R}^3 : \mathcal{C}(\boldsymbol{x}) \wedge X = 0 \right\}. \tag{10}$$

A similar expression can be found for the inner product nullspace. The conformal points are the basic building blocks to construct other geometric primitives in their OPNS representation. The relevant primitives for this work are lines

$$L = P_1 \wedge P_2 \wedge \boldsymbol{e}_\infty, \tag{11}$$

which can be constructed from two points and a point at infinity, planes

$$E = P_1 \wedge P_2 \wedge P_3 \wedge \boldsymbol{e}_\infty, \tag{12}$$

which can be constructed from three points and a point at infinity and spheres

$$S = P_1 \wedge P_2 \wedge P_3 \wedge P_4, \tag{13}$$

which can be constructed from four points.

Rigid body transformations in CGA are achieved using motors M, which are exponential mappings of dual lines, i.e. bivectors (essentially, the screw axis of the motion). Note that motors can be used to transform any object in the algebra, i.e. they can directly be used to transform the previously introduced points, lines, planes and spheres, by a sandwiching operation

$$X' = M X \widetilde{M}, \tag{14}$$

where is $\widetilde{M}$ is the reverse of a motor.

The forward kinematics of serial kinematic chains can be found as the product of motors, i.e.

$$M(\boldsymbol{q}) = \prod_{i=1}^{N} M_i(q_i) = \prod_{i=1}^{N} \exp(q_i B_i), \tag{15}$$

where $\boldsymbol{q}$ is the current joint configuration and B_i are screw axes of the joints. The geometric Jacobian $\boldsymbol{\mathcal{J}}^G(\boldsymbol{q}) \in \mathbb{B}^{1\times N} \subset \mathbb{G}_{4,1}^{1\times N}$ is a bivector valued multivector matrix and can be found as

$$\boldsymbol{\mathcal{J}}_G = \begin{bmatrix} B_1' & \dots & B_N' \end{bmatrix}, \tag{16}$$

where the bivector elements can be found as

$$B_i' = \prod_{j=1}^{i} M_j(q_j) B_i \prod_{j=1}^{i} \widetilde{M}_j(q_j). \tag{17}$$

Twists $\mathcal{V}$ and wrenches $\mathcal{W}$ are also part of the algebra and hence both can be transformed in the same manner as the geometric primitives using Equation (14). Note that, contrary to classic matrix Lie algebra, no dual adjoint operation is needed to transform wrenches. There is, however, still a duality relationship between twists and wrenches, which can be found via multiplication with the conjugate pseudoscalar $I_c = I\boldsymbol{e}_0$ [70]. Both twists and wrenches are bivectors and the space of wrenches can be found as

$$\mathcal{W} \in \text{span}\{\boldsymbol{e}_{23}, \boldsymbol{e}_{13}, \boldsymbol{e}_{12}, \boldsymbol{e}_{01}, \boldsymbol{e}_{02}, \boldsymbol{e}_{03}\}. \tag{18}$$

The inner product of twists and wrenches $\mathcal{V} \cdot \mathcal{W} = -p$ yields a scalar, where p is the power of the motion. Similarly, the inner product of a screw axis and a wrench $B \cdot \mathcal{W} = -\tau$ yields a torque τ, which we will use for the task-space impedance control in this article.

IV. METHOD

We present our closed-loop tactile ergodic coverage method in three parts: (i) surface preprocessing; (ii) tactile coverage; and (iii) robot control. The surface preprocessing computes the quantities that need to be calculated only once when the surface is captured. Tactile coverage generates the motion commands for the virtual coverage agent using the precomputed quantities from the surface preprocessing and the robot controller tracks the generated motion commands with a manipulator using impedance control.

A. Problem Statement

We formulate a tactile ergodic controller that covers a target spatial distribution on arbitrary surfaces. Similar to HEDAC, we propagate the information encoding the coverage objective by solving the diffusion equation on the manifold $\mathcal{M}$

$$\frac{\partial}{\partial t}\boldsymbol{u} = \boldsymbol{\Delta}_{\mathcal{M}}\boldsymbol{u}, \tag{19}$$

where we refer to $\boldsymbol{u}$ as the potential field. Note that we use here $\boldsymbol{\Delta}_{\mathcal{M}}$, which generalizes the Laplacian for Euclidean spaces $\boldsymbol{\Delta}$ to non-Euclidean manifolds $\mathcal{M}$. This operator $\boldsymbol{\Delta}_{\mathcal{M}}$ is also known as Laplace-Beltrami operator but for conciseness we will use the term Laplacian. In general, our coverage domains are curved surfaces (i.e. 2-manifolds). Here, we capture the underlying manifold $\mathcal{M}$ as a point cloud $\mathcal{P}$ composed of $n_{\mathcal{P}}$ points using an RGB-D camera

$$\mathcal{P} := \left\{ (\boldsymbol{x}_i, \boldsymbol{c}_i) \;\middle|\; \begin{array}{l} \boldsymbol{x}_i \in \mathbb{R}^3,\ \boldsymbol{c}_i \in \{0, \ldots, 255\}^3 \\ \text{for } i = 1, \ldots, n_{\mathcal{P}} \end{array} \right\}, \tag{20}$$

where $\boldsymbol{x}_i$ is the position of the i-th surface point in Euclidean space and $\boldsymbol{c}_i$ is the vector of RGB color intensities. We assume there is a processing pipeline (i.e., such as [20], [47], [71]) which maps the point positions and colors to the probability mass p_i of the spatial distribution encoding the coverage objective. Accordingly, our coverage target becomes a discrete spatial distribution $p(\boldsymbol{x}_i) = p_i$ on the point cloud $\mathcal{P}$.

In order to solve (19) on irregular and discrete domains, such as point clouds, we discretize the problem in space and time. Hence, we use $u_{i,t}$ to denote the value of the potential field at the i-th point at the t-th timestep. We omit the subscript i if we refer to all points.

B. Surface Preprocessing

First, we compute the spatial discretization of the Laplacian $\boldsymbol{\Delta}_{\mathcal{M}}$. Note that there are various approaches for discretizing the Laplacian on point clouds [49]–[52]. In this work, we follow the approach presented in [52] and show a simplified version of it here, but refer the readers to the original work for more details. Using this method, the discrete Laplacian is represented by the matrix $\boldsymbol{L} \in \mathbb{R}^{n_{\mathcal{P}} \times n_{\mathcal{P}}}$

$$\boldsymbol{L} = \boldsymbol{M}^{-1}\boldsymbol{C}, \tag{21}$$

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 the 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 a boundary and its boundary condition is set as zero-Neumann, i.e., $\nabla \boldsymbol{u} \cdot \boldsymbol{n} = 0$ where $\boldsymbol{n}$ is the outward normal of the boundary.

Next, we discretize the diffusion equation (19) in time and insert the discrete Laplacian $\boldsymbol{L}$. Using the backward Euler method, we obtain the implicit equation

$$\frac{1}{\delta_t}(\boldsymbol{u}_t - \boldsymbol{u}_0) = \boldsymbol{L}\boldsymbol{u}_t, \tag{22}$$

which is stable for any timestep δ_t. Then, combining Equations (21) and (22) and solving for $\boldsymbol{u}_t$, we obtain the linear system

$$\boldsymbol{u}_t = (\boldsymbol{M} - \delta_t\, \boldsymbol{C})^{-1}\boldsymbol{M}\boldsymbol{u}_0. \tag{23}$$

Note that solving (23) requires inverting a large sparse matrix, which might be computationally expensive depending on the size of the point cloud and requires the timestep 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 arbitrary manifolds. Note that the Fourier series are the eigenfunctions of the Laplacian $\boldsymbol{\Delta}$ 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

$$\boldsymbol{C}\boldsymbol{\phi}_m = \lambda_m \boldsymbol{M}\boldsymbol{\phi}_m, \tag{24}$$

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. Furthermore, the eigenvectors 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

$$\boldsymbol{u}^{\boldsymbol{\phi}} = \boldsymbol{\Phi}^\top \boldsymbol{M} \boldsymbol{u}^{\boldsymbol{x}}. \tag{25}$$

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 $\dot{x}(t) = -cx(t)$ is given by $x(t) = e^{-ct}x(0)$, where c is a constant and $x(0)$ is the initial value. Therefore, the solution of the system of ODEs in the spectral domain is given in matrix form as

$$\boldsymbol{u}_t^{\boldsymbol{\phi}} = \left[\begin{array}{ccc} e^{-\lambda_1 \delta_t} & \dots & e^{-\lambda_m \delta_t} \end{array} \right]^\top \odot \boldsymbol{u}_0^{\boldsymbol{\phi}}, \tag{26}$$

where $\odot$ denotes the Hadamard product. We observe from (26) 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

$$\boldsymbol{u}^{\boldsymbol{x}} = \boldsymbol{\Phi} \boldsymbol{u}^{\boldsymbol{\phi}}. \tag{27}$$

We can combine (25), (26) and (27) into a unified spectral scheme

$$\boldsymbol{u}_t = \Phi \left[\begin{array}{ccc} e^{-\lambda_1 \delta_t} & \dots & e^{-\lambda_m \delta_t} \end{array} \right]^\top \odot \left(\Phi^\top \boldsymbol{M} \boldsymbol{u}_0 \right). \tag{28}$$

We omit the superscripts when working on the point cloud for brevity. Note that δ_t 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 $\alpha > 0$ and embed it into the timestep calculation

$$\delta_t = \alpha h^2. \tag{29}$$

Accordingly, we can control the diffusion behavior independently of the point cloud size. Increasing α results in longer diffusion times and attenuates the high-frequency spatial features (see (26) for details). This corresponds to a more global coverage [56]. Conversely, decreasing α results in shorter diffusion times, which leads to preserving the high-frequency spatial features, hence more local coverage behavior.

Note that the Laplacian is determined completely by the connectivity on the local tangent space and the distance between these connected points. Therefore, it is invariant to distance preserving (i.e., isometric) transformations such as rigid body motion or deformation without stretching. Accordingly, 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 p_i changes.

C. Tactile Ergodic Coverage

We model the actual coverage tool/sensor as a compliant virtual coverage agent shaped as a disk with radius r_a. Notably, one can represent arbitrary tool/sensor footprints as a combination of disks [56]. We position our agent at the end-effector of our manipulator. Thus, for a given kinematic chain and joint configuration $\boldsymbol{q}$, we can use the forward kinematics to compute the position of our agent as a conformal point P_a

$$P_a = M(\boldsymbol{q}) \boldsymbol{e}_0 \widetilde{M}(\boldsymbol{q}). \tag{30}$$

Since the point cloud is discrete and the agent should move continuously on the surface, we project our agent P_a and its footprint to the closest local tangent space on the point cloud.

1) Local Tangent Space and Coverage Computation: Given the agent's position P_a, we first compute the closest tangent space on the point cloud. For that, we query a K-D tree $\mathcal{T}(\mathcal{P})$ for the points $\boldsymbol{x}_i \in \mathcal{P}$ that are within the radius r_a of the agent. Then, we compute the conformal embeddings P_i of the neighboring Euclidean points $\boldsymbol{x}_i$ using (9). We refer to the set composed of points P_i as the local neighborhood. Then, we fit a tangent space to the local neighborhood by minimizing the classical least squares objective

$$\min \sum_{i=1}^{n_N} (P_i \cdot X^*)^2, \tag{31}$$

where X^* is the dual representation of either a plane or a sphere and the inner product $\cdot$ is a distance measure. In CGA, planes can be seen as limit cases of spheres, i.e. planes are spheres with infinite radius. This is also easy to observe by looking at Equations (12) and (13) which construct these geometric primitives. Note that fitting a local tangent sphere with the radius determined by the local curvature would always result in smaller or equal residuals than fitting a plane.

It has been shown in [72] that the solution to the least squares problem given in (31) is the eigenvector corresponding to the smallest eigenvalue of the 5×5 matrix

$$b_{j,k} = \sum_{i=1}^{n_N} w_{i,j} w_{i,k}, \tag{32}$$

where

$$w_{i,k} = \begin{cases} p_{i,k} & \text{if } k \in \{1,2,3\} \\ -1 & \text{if } k = 4 \\ -\frac{1}{2}\boldsymbol{p}_i^2 & \text{if } k = 5. \end{cases} \tag{33}$$

Using the five components v_i of this eigenvector we can find the geometric primitive as

$$X = \left(v_0 \boldsymbol{e}_0 + v_1 \boldsymbol{e}_1 + v_2 \boldsymbol{e}_2 + v_3 \boldsymbol{e}_3 + v_4 \boldsymbol{e}_\infty \right)^*. \tag{34}$$

Note that if X is a plane then $v_0 = 0$, otherwise X is a sphere. Next, we want to project P_a to X by using the general subspace projection formula of CGA

$$P_{pair} = \big((P_a \wedge \boldsymbol{e}_\infty) \cdot X \big) X^{-1}. \tag{35}$$

Here we first construct the pointpair $P_a \wedge \boldsymbol{e}_\infty$, where $\boldsymbol{e}_\infty$ corresponds to the point at infinity. $P_a \wedge \boldsymbol{e}_\infty$ is also called a flat point. Note that the projection essentially amounts to first constructing the dual line $(P_a \wedge \boldsymbol{e}_\infty) \cdot X$ that passes through the point P_a and is orthogonal to X, then intersecting this line with the primitive X.

If X is a sphere, then the intersection of the line and the sphere will result in two points on the sphere. If X is a plane, it will result in another flat point, i.e. one point on the plane and one at infinity. In any case, we can retrieve the closer one to the agent position P_a using the split operation

$$P_a' = \text{split}\left[P_p \right]. \tag{36}$$

Here, P_a' is the projected agent position on the tangent space X. Next, we compute our agent's footprint (i.e., instantaneous coverage) by projecting its surface to the point cloud. If the target surface was flat, all the points within the radius r_a of our agent P'_a would be covered by the footprint. However, in the general case, both the tool and the surface can be curved and deformable. For simplicity, we assume that the surface is rigid and it deforms the tool with a constant bending radius. We use the radius of the local tangent sphere that we computed using CGA as an approximation for the bending radius. Accordingly, we can quantify the error of the local tangent space approximation for the i-th neighbor P_i by the normalized residuals e_i of the least squares computation (31). We encode this approximation error into the footprint by weighting the i-th neighbor by the Gaussian kernel $\varphi(r)$ using the normalized residuals $r_i = e_i / \max(\boldsymbol{e})$

$$\varphi(r_i) = \exp\left(-\varepsilon^2 r_i^2\right), \tag{37}$$

where the hyperparameter $\varepsilon > 0$ controls the coverage falloff. Next, we plug the Gaussian kernel weighted footprint in (3) for computing the coverage $\boldsymbol{c}_t$, which then allows us to calculate the virtual source term $\boldsymbol{s}_t$ using (1).

The coverage objective at the t-th timestep is embedded in the source term $\boldsymbol{s}_t$. Therefore, we set it as the initial condition of the diffusion equation (19), i.e., $\boldsymbol{u}_0 = \boldsymbol{s}_t$ and use either the implicit (23) or spectral (28) formula to diffuse the resulting potential field. Note that at each iteration of the tactile coverage loop, we solve an independent diffusion problem starting from $t = 0$.

2) Gradient of the Diffused Potential Field: We guide the coverage agent using the gradient of the diffused potential field as the acceleration command

$$\ddot{P}'_a = \nabla \boldsymbol{u}_{P'_a,t}, \tag{38}$$

where $\nabla \boldsymbol{u}_{P'_a,t}$ denotes the gradient of the diffused potential field at the projected agent position P'_a. However, computing the gradient on the point cloud is more involved than a regular grid or a mesh. Recall that in Section IV-C1, we already computed the projected agent position P'_a, the local neighborhood and the tangent space X^*. As the first step, we compute the tangent plane $E_{a,t}$ at P'_a, namely

$$E_{a,t} = L^*_{a,\perp} \wedge P'_a \wedge \boldsymbol{e}_\infty, \tag{39}$$

using the line $L_{a,\perp}$, which is orthogonal to the surface and passes through P'_a. It is found by wedging the dual primitive X with P'_a to infinity with

$$L_{a,\perp} = X^* \wedge P'_a \wedge \boldsymbol{e}_\infty. \tag{40}$$

Then, we project the points P_i in the local neighborhood to the tangent plane $E_{a,t}$ using (35) and (36), by setting $E_{a,t}$ as the primitive X. Next, we use the values of the potential field at the neighbor locations as the height $\boldsymbol{h}_i = \boldsymbol{u}_{i,t}$ of a second surface from the tangent plane. Then, we fit a 3-rd degree polynomial to this surface as shown by using the weighted least squares objective

$$\hat{\boldsymbol{A}} = \arg\min_{\boldsymbol{A}} \operatorname{tr}\left((\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{A})^\top \boldsymbol{W} (\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{A})\right), \tag{41}$$

with the diagonal weight matrix $\boldsymbol{W}$

$$\boldsymbol{W} = \operatorname{diag}\left(\varphi(r_1), \varphi(r_1), \ldots \varphi(r_m)\right), \tag{42}$$

whose entries are given by the Gaussian kernel (37). One can refer to [73] for the details. 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 2.

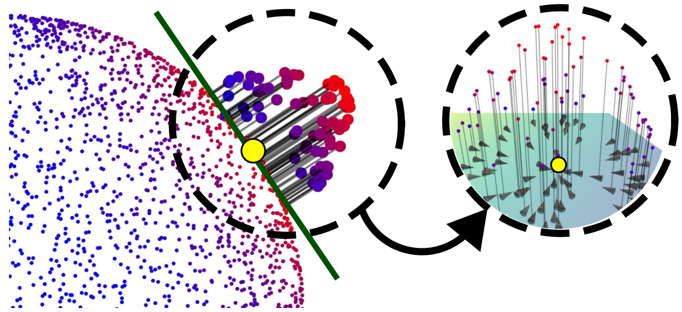

Fig. 2: Blue-red points show the value of the potential field $\boldsymbol{u}_t$ on the pointcloud $\mathcal{P}$ and the yellow point is the projected agent position P'_a. We also project the agent's neighbors P_i to the tangent plane $E_{a,t}$, shown in green. Next, we use the height function $\boldsymbol{h}_i = \boldsymbol{u}_{i,t}$ 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 $\boldsymbol{\nabla u}_{i,t}$, as shown with arrows in the detail view.

D. Robot Control

There are several aspects that the control of the physical robot needs to achieve. The first is to track the virtual coverage agent on the target surface, while keeping the end-effector normal to the surface. The second is to exert a desired force on the surface. To do so, we design a task-space impedance controller while further exploiting geometric algebra for efficiency and compactness. The control law is of the following form

$$\boldsymbol{\tau} = -\boldsymbol{\mathcal{J}}^\top \cdot \mathcal{W}, \tag{43}$$

where $\boldsymbol{\mathcal{J}} \in \mathbb{B}^{1\times N} \subset \mathbb{G}^{1\times N}_{4,1}$ is the Jacobian multivector matrix with elements corresponding to bivectors, $\mathcal{W}$ is the desired task-space wrench and $\boldsymbol{\tau}$ are the resulting joint torques. Before composing the final control law, we will explain its components individually.

1) Surface Orientation: From Equation (40), we obtained a line $L_{a,\perp}$ that is orthogonal to the surface that we wish to track. In [74], it was shown how the motor between conformal objects can be obtained. We use this formulation to find the motor between the target orthogonal line and the line that corresponds to the z-axis of the end-effector of the robot in its current configuration, which is found as

$$L_{ee} = M(\boldsymbol{q})(\boldsymbol{e}_0 \wedge \boldsymbol{e}_3 \wedge \boldsymbol{e}_\infty)\widetilde{M}(\boldsymbol{q}). \tag{44}$$

Then, the motor $M_{L_{ee}L_{a,\perp}}$, which transforms L_{ee} into $L_{a,\perp}$ can be found as

$$M_{L_{ee}L_{a,\perp}} = \frac{1}{C}\left(1 + L_{a,\perp} L_{ee}\right), \tag{45}$$

where C is a normalization constant. Note that C does not simply correspond to the norm of $1 + L_{a,\perp} L_{ee}$, but requires a more involved computation. We therefore omit its exact computation here for brevity and refer readers to [74].

We can now use the motor $M_{L_{ee}L_{a,\perp}}$ in order to find a control command for the robot via the logarithmic map of motors, i.e.

$$\mathcal{V}_{L_{a,\perp}} = \log\left(M_{L_{ee}L_{a,\perp}}\right). \tag{46}$$

Of course, if the lines are equal, $M_{L_{ee}L_{a,\perp}} = 1$ and consequently $\mathcal{V}_{L_{a,\perp}} = 0$. Note that $\mathcal{V}_{L_{a,\perp}}$ is still a command in task space (we will explain how to transform it to a joint torque command once we have derived all the necessary components).

Another issue is that algebraically, $\mathcal{V}_{L_{a,\perp}}$ corresponds to a twist, not a wrench. Hence, we need to transform it accordingly. From physics, we know that twists transform to wrenches via an inertial map, which we could use here as well. In the context of control, this inertia tensor is, however, a tuning parameter and does not actually correspond to a physical quantity. Thus, in order to simplify the final expression, we will use a scalar matrix valued inertia, instead of a geometric algebra inertia tensor and choose to transform the twist command to wrench command purely algebraically. As it has been shown before, this can be achieved by the conjugate pseudoscalar $I_c = \boldsymbol{I}\boldsymbol{e}_0$ [70]. It follows that

$$\mathcal{W}_{L_{a,\perp}} = \mathcal{V}_{L_{a,\perp}} I_c, \tag{47}$$

and $\mathcal{W}_{L_{a,\perp}}$ now algebraically corresponds to a wrench.

2) Target Surface Force: Since this article describes a method for tactile surface coverage, the goal of the robot control is to not simply stay in contact with the surface, but to actively exert a desired force on the surface. First of all, we denote the current measured wrench as $\mathcal{W}_m(t)$ and the desired wrench as $\mathcal{W}_d$. Both are bivectors as defined by Equation (18). We use $\mathcal{W}_d$ w.r.t. end-effector in order to make it more intuitive to define. Hence, we need to transform $\mathcal{W}_m(t)$ to the same coordinate frame, i.e.

$$\mathcal{W}'_m(t) = \widetilde{M}(\boldsymbol{q})\mathcal{W}_m(t)M(\boldsymbol{q}). \tag{48}$$

In order to achieve the desired, we simply apply a standard PID controller in wrench space, i.e.

$$\mathcal{W}_C = \boldsymbol{K}_{p,\mathcal{W}}\mathcal{W}_e + \boldsymbol{K}_{i,\mathcal{W}}\int_0^{\top}\mathcal{W}_e(\tau)\mathrm{d}\tau + \boldsymbol{K}_{d,\mathcal{W}}\frac{\mathrm{d}}{\mathrm{d}t}\mathcal{W}_e(t), \tag{49}$$

where the wrench error is

$$\mathcal{W}_e(t) \quad = \quad \mathcal{W}_d - \mathcal{W}'_m(t), \tag{50}$$

where $\boldsymbol{K}_{p,\mathcal{W}}, \boldsymbol{K}_{i,\mathcal{W}}$ and $\boldsymbol{K}_{d,\mathcal{W}}$ are the corresponding gain matrices, and $\mathcal{W}_C$ is the resulting control wrench.

3) Task-Space Impedance Control: Recalling the control law from Equation (43), we now collect the terms from the previous subsections into a unified task-space impedance control law. We start by looking in more detail at the Jacobian $\boldsymbol{\mathcal{J}}$. Previously, we mentioned that we are using the current end-effector motor as the reference, hence, we require the Jacobian to be computed w.r.t. that reference. This is therefore not the geometric Jacobian that was presented in Equation (16), but a variation of it. The end-effector frame geometric Jacobian $\boldsymbol{\mathcal{J}}_G^{ee}$ can be found as

$$\boldsymbol{\mathcal{J}}_G^{ee} = \begin{bmatrix} B_1^{ee} & \dots & B_N^{ee} \end{bmatrix}, \tag{51}$$

where the bivector elements can be found as

$$B_i^{ee} = \widetilde{M}_i^{ee}(\boldsymbol{q})B_i M_i^{ee}, \tag{52}$$

with

$$M_i^{ee} = \prod_{j=N}^{i} M_i(q_i). \tag{53}$$

Hence, the relationship between $\boldsymbol{\mathcal{J}}_G$ and $\boldsymbol{\mathcal{J}}_G^{ee}$ can be found as

$$\boldsymbol{\mathcal{J}}_G^{ee} = \widetilde{M}(\boldsymbol{q})\boldsymbol{\mathcal{J}}_G M(\boldsymbol{q}). \tag{54}$$

The wrench in the control law is composed of the three wrenches that we defined in the previous subsections. As commonly done, we add a damping term that corresponds to the current end-effector twist and as before, we transform it to an algebraic wrench, i.e.

$$\mathcal{W}_{\mathcal{V}} = \boldsymbol{\mathcal{J}}_G^{ee}\dot{\boldsymbol{q}}\boldsymbol{e}_{0\infty}. \tag{55}$$

With this, we now have everything in place to compose our final control law as

$$\boldsymbol{\tau} = -\boldsymbol{\mathcal{J}}_G^{ee,\top}\cdot\left(\boldsymbol{K}_{L_{a,\perp}}\mathcal{W}_{L_{a,\perp}} - \boldsymbol{D}_{\mathcal{V}}\mathcal{W}_{\mathcal{V}} + \mathcal{W}_C\right), \tag{56}$$

where $\boldsymbol{K}_{L_{a,\perp}}$ is a stiffness and $\boldsymbol{D}_{\mathcal{V}}$ a damping gain.

V. EXPERIMENTS

Our experimental setup comprises a BotaSys SensOne 6-axis force torque (F/T) sensor attached to the wrist of a 7-axis Franka Emika robot manipulator and a custom 3-D printed part attached to the F/T sensor. The custom part interfaces an Intel Realsense D415 depth camera and a sponge at its tip. We consider the sponge's center point to be the coverage agent's position P_a. Before the operation, we perform extrinsic calibration of the camera to combine the depth and RGB feeds from the camera and to obtain its transformation with respect to the robot joints. Additionally, we calibrate the F/T sensor to compensate for the weight of the 3-D printed part and the camera. We show the experimental setup on the left of Figure 1.

A. Implementation Details

The pipeline of our tactile ergodic coverage method consists of three modules: (i) surface acquisition, (ii) surface coverage and (iii) robot control. Figure 3 summarizes the information flow between the components.

1) Surface Acquisition: The surface acquisition node is responsible for collecting the point cloud and performing preprocessing operations described in Section IV-B. We use *scipy*[1] for the nearest neighbor queries and for solving the eigenproblem in (24). The matrices $\boldsymbol{C}$ and $\boldsymbol{M}$ composing the discrete Laplacian in (21) are computed with the *robust_laplacian* package[2] [52].

2) Surface Coverage: The surface coverage node performs the computations based on the procedure given in Section IV-C. It uses the information provided by the surface acquisition node and produces the target *line* for the robot control node.

[1]https://scipy.org

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

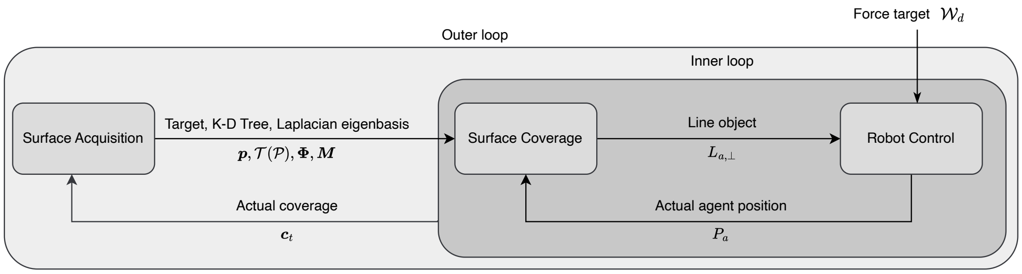

Fig. 3: Information flow between the three components. The pipeline is composed of an outer loop responsible for controlling the coverage progress with the feedback from the camera, whereas the inner loop compensates for the mismatch due to the robot dynamics.

3) Robot control: On a high level, the robot control can be seen as a state machine with three discrete states. The first two states are essentially two pre-recorded joint positions in which the robot is waiting for other parts of the pipeline to be completed. One of these positions corresponds to the picture-taking position, i.e., a joint position where the camera has the full object in its frame and the point cloud can be obtained. The robot is waiting in this position until the point cloud has been obtained, afterwards it changes its position to hover shortly over the object. In this second position, it is waiting for the computation of the Laplacian eigenfunctions to be completed, such that the coverage can start. The switching between those two positions is achieved using a simple joint impedance controller.

The third, and most important, state is when robot is actually controlled to be in contact with the surface and to follow the target corresponding to the coverage agent. This behaviour is achieved using the controller that we described in Section IV-D. The relevant parameters, that were chosen empirically for the real-world experiments, are the stiffness and damping of the line tracking controller, i.e. $K_{L_{a,\perp}} = \mathrm{diag}(30, 30, 30, 750, 750, 300)$ and $D_{\mathcal{V}} = \mathrm{diag}(10, 10, 10, 150, 150, 50)$, as well as the gains of the wrench PID controller, i.e. $K_{p,\mathcal{W}} = 0.5$, $K_{i,\mathcal{W}} = 5$ and $K_{d,\mathcal{W}} = 0.5$. The controller has been implemented using our open-source geometric algebra for robotics library *gafro*[3] that we first presented in [75]. Note that in some cases, matrix-vector products of geometric algebra quantities have been used for the implementation, where the mathematical structure of the geometric product actually simplifies to this, which can be exploited for more efficient computation.

B. Simulated Experiments

1) Computation Performance: In order to assess the computational performance, we investigated the two main operations of our method: (i) preprocessing by solving either the eigenproblem (24) or matrix inversion in (23) (ii) integrating the diffusion at runtime using either the spectral (28) or implicit (23) formulations. In this experiment, we used the Stanford Bunny as the reference point cloud and performed voxel filtering to set the point cloud resolution. We present the results for the preprocessing in Figure 4 and for the runtime in Figure 5.

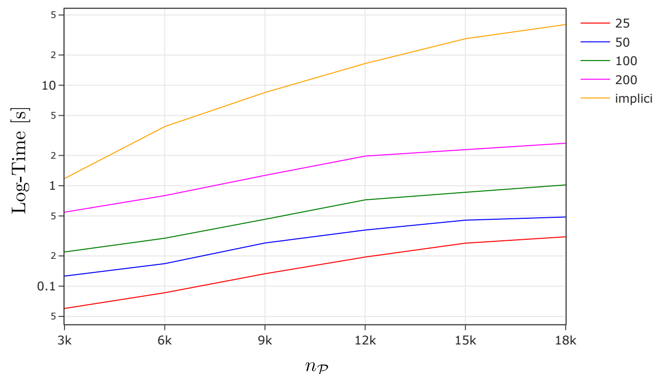

Fig. 4: Computational complexity of the preprocessing step for different $n_{\mathcal{P}}$ and n_M. Legend shows n_M values. The time axis is logarithmic and the legend shows n_M values.

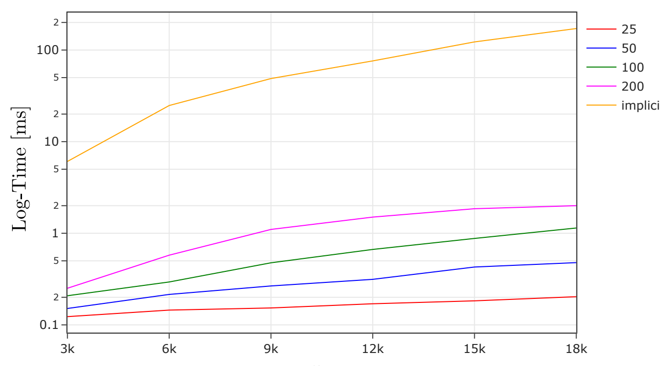

Fig. 5: Computational complexity of integrating the diffusion equation at runtime for different $n_{\mathcal{P}}$ and n_M. The time axis is logarithmic and the legend shows n_M values.

2) Coverage Performance: We tested the coverage performance in a series of kinematic simulations. As the coverage metric, we used the normalized ergodicity over the target distribution, which compares the time-averaged statistics of agent trajectories to the target distribution

$$\varepsilon_t = \frac{\| \max{(\boldsymbol{p} - \boldsymbol{c}_t, 0)} \|_2}{\sum_{i=1}^{n_{\mathcal{P}}} p_i}. \tag{57}$$

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

We ran the experiments for three different objects: a partial point cloud of the Stanford Bunny and two point clouds of a cup and a plate and their target distributions that we collected using the RGB-D camera. For the Stanford bunny, we projected an 'X' shape as the target distribution. For each object, we sampled ten different initial positions for the coverage agent and kinematically simulated the coverage using different numbers of eigencomponents $n_M = 25, 50, 100, 200$ and diffusion timestep scalar $\alpha = 1, 5, 10, 50, 100$. Since the plate is larger compared to the Bunny and the cup, we used a larger agent radius $r_a = 15$ $[mm]$ for the plate and a smaller value $r_a = 7.5$ $[mm]$ for the cup and the Bunny. The other parameters that we kept constant in all of the experiments are $\ddot{x}_{\text{max}} = 3$ $[mm/s^2]$, $\dot{x}_{\text{max}} = 3$ $[mm/s]$. We selected six representative experiment runs to show the coverage performance qualitatively, and present them in Figure 6.

We show the quantitative results with respect to n_M and α in Figures 7 and 8, respectively. Note that, in order to better show performance trend in these plots, we have excluded parameter combinations leading to failure cases. We will discuss those in Section VI.

As the last experiment, we chose the best-performing pair (n_M, α) and show the time evolution of the coverage performance for different objects in Figure 9.

C. Real-world Experiment

In the real-world experiments, we tested the whole pipeline presented in Section V-A. We used three different kitchen utensils (plate, bowl, and cup) with different target distributions (shapes, RLI, X). For these experiments, we fixed the objects to the table so that they could not move when the robot was in contact. At the beginning of the experiments, we moved the robot to a predefined joint configuration that fully captured the target distribution. Since we collected the point cloud data from a single image frame, our method only had access to a partial and noisy point cloud. We summarize the results of the real-world experiments in Figure 10 and share all the recorded experiment data and the videos on the accompanying website.

D. Comparisons

We present the first tactile ergodic coverage method in the literature that works on curved surfaces. Therefore, there are no methods that we can directly compare to quantitatively. For this reason, we selected three related state-of-the-art methods and compared them to our method qualitatively. As the first method, we selected the finite element based HEDAC planner [46], since it is the only other ergodic control approach working on curved surfaces. For the tactile interaction aspect, we selected two methods, the unified force-impedance control [76] and the sampling-based informative path planner [20]. We specified six criteria for comparison and summarized the results in Table I.

VI. DISCUSSION

A. Computational Performance

We investigated the computational performance of our method for the preprocessing and for the runtime.

The preprocessing step is only required, when the robot sees an object for the first time or when the object undergoes a non-isometric transformation. First thing to note from Figure 4 is that computing the eigenbasis is significantly faster than inverting the large sparse matrix. Secondly, the advantage of the spectral approach becomes more significant as the number of points increases. This is because the computational complexity of the spectral approach is linear $\mathcal{O}(n_{\mathcal{P}} n_M)$ with the number of points, whereas the matrix inversion of the implicit solution has quadratic complexity $\mathcal{O}(n_{\mathcal{P}}^2)$.

If we compare our method with the state-of-the-art in ergodic coverage on curved surfaces [46], our preprocessing step is significantly faster. They reported a computation time of $19.7\,\mathrm{s}$ for a mesh with 2315 points using a finite-element-based method. In contrast, our method takes $278\,\mathrm{ms}$ for a point cloud with ≈ 3000 points with $n_M = 100$. Therefore, in comparison, our method promises an increase in computation speed of more than 90 times. Note that, as the number of points increases, our gains in computation time become even more significant due to the difference in the computational complexity of the spectral and implicit formulations as mentioned above.

As Figure 5 shows the spectral approach also results in a significant performance increase at runtime. The implicit solution is also efficient in runtime, since it reduces to matrix-vector multiplication after inverting the sparse matrix at the preprocessing step. Nevertheless, the spectral formulation is still significantly faster than the implicit formulation, especially for large point clouds.

Obviously, an unnecessarily large eigenbasis for small point clouds, i.e. $n_M \rightarrow n_{\mathcal{P}}$, would cause the spectral approach to be slower than the implicit one.

B. Coverage Performance

A close investigation of the failure scenarios in Figure 6 revealed that they stem from the bad coupling of the parameters and from an initialization of the agent far away from the source. If the agent is not far away from the source, setting low values for α might actually lead to desirable properties such as prioritizing local coverage which would in turn minimize the distance traveled during coverage. Hence, for getting the best behavior, α can be set adaptively or sequentially. For instance, it is better to use high α values at the start for robustness to bad initializations and to decrease it as the coverage advances to prioritize local coverage and to increase the performance.

We measured the effect of our method parameters on the coverage performance in Figures 7 and 8. Interestingly, the parameters influencing the agent's speed, i.e. $\dot{\boldsymbol{x}}_{max}$, n_M and α, have a coupled effect on the coverage performance in some of the scenarios. The first thing to note here is that the value of the α is lower-bounded by the speed of the coverage agent $\dot{\boldsymbol{x}}_{max}$. Otherwise the method cannot guide the agent since it moves faster than the diffusion. For instance, we observe from Figure 6 a) and f) that with a diffusion coefficient $\alpha = 1$, the source information does not propagate fast enough to the agent if it is too far from the source. Even for a small eigenbasis $n_M \leq 50$ and moderate diffusion coefficient values $1 < \alpha \leq 10$, it still results in a low coverage performance $\boldsymbol{\varepsilon}_t > 0.5$.

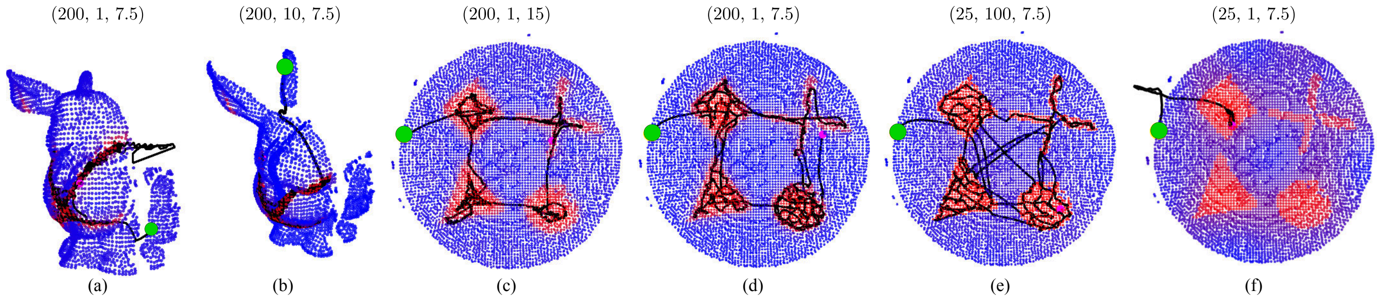

Fig. 6: Qualitative results of the coverage experiments showcasing the effect of different parameters. The red points designate the spatial target distribution $p_i > 0$. The agent starts at the green point, the trajectory is shown in black, and the final position after 1000 timesteps is shown with the purple point. The tuples given on top of the figures show the parameters $n_{\mathcal{K}}$, α, and r_a of the experiments. We provide the interactive point clouds and the experiment data on our website.

TABLE I: Comparison of the proposed method with state-of-the-art methods.

Method	Domain	Approach	Online	Purpose	Multiscale	Multisetup [a]
Finite element-based HEDAC [46]	Mesh	Planning	No	Visual Inspection	Yes	No
Sampling-based Planner [20]	Mesh	Planning	Yes	Tactile Coverage	No	Yes
Unified Force-Impedance Control [76]	None	Control	Yes	Surface Exploration	No	No
Tactile Ergodic Control (Ours)	Point Cloud [b]	Control	Yes	Tactile Coverage	Yes	No

[a]Multisetup used by [20] refers to planning the configuration of the target object to reach otherwise unreachable regions.

[b]Since point clouds are the most general representation, our methoud can seamlessly be used on grids/meshes with only minor changes to the computation of the discrete Laplacian.

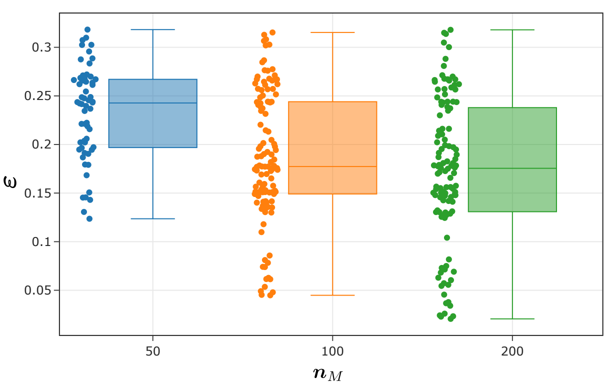

Fig. 7: Coverage performance measured by the ergodic metric ε_t (57) with respect to n_M used in the spectral formulation (26).

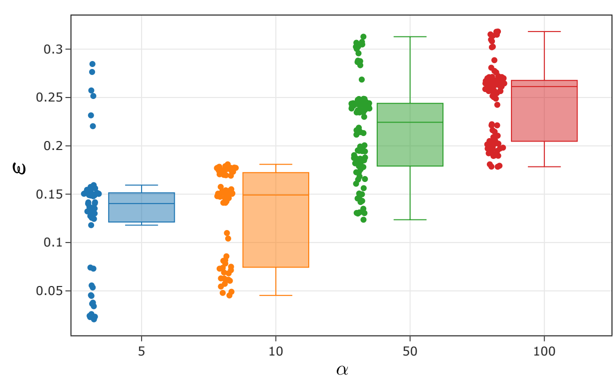

Fig. 8: Coverage performance measured by the normalized ergodic metric ε_t (57) with respect to the parameter α.

On the contrary, if the eigenbasis is chosen to be sufficiently large $n_M \geq 100$, we have more freedom in choosing α.

With this in mind, we removed the infeasible parameter combinations $(n_M = 50, \alpha = \{5, 10\})$ from the experiment results in Figures 7 and 8 to better observe the performance trend for n_M and α. It is easy to see that increasing n_M results in increased performance and higher freedom in choosing α. However, this benefit becomes marginal after $n_M \geq 100$. Therefore, choosing $n_M = 100$ becomes a good trade-off between coverage performance and computational complexity. This observation is in line with the value of $n_M = 128$ reported in [47].

In Figure 8, however, we observed minor differences in performance for different α. Considering the spread and the mean, choosing $\alpha = 10$ would be a good fit for most scenarios. Nevertheless, we must admit that the ergodic metric falls short in distinguishing the most significant differences between α values. Hence, the qualitative performance shown in Figure 6 becomes much more explanatory. The first thing to note here is that the lower values of α result in more local coverage, whereas higher values lead to prioritizing global coverage. Accordingly, the tuning of this parameter depends on the task itself. For example, suppose the goal is to collect measurements from different modes of a target distribution as quickly as possible, in which case we would recommend using $\alpha > 50$. On the other hand, if the surface motion is costly, because for example, the surface is prone to damage, moving less frequently between the modes can be achieved by setting $5 < \alpha < 50$.

In scenarios where the physical interactions are complex, stopping the coverage prematurely and observing the actual

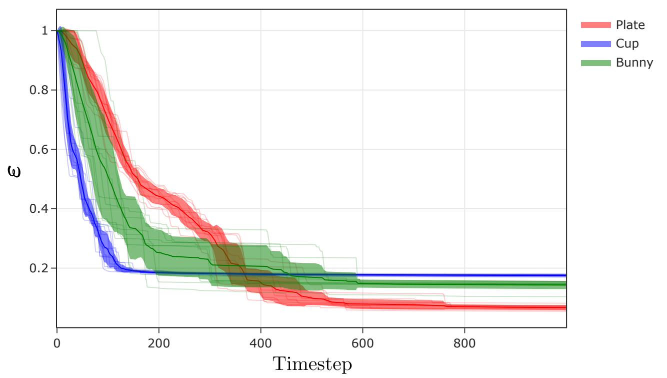

Fig. 9: Time evolution of the ergodic metric (57) for three different objects with $n_M = 200$ and $\alpha = 10$. The semi-transparent lines show ten different experiment runs, the center line shows the mean, and the shaded regions correspond to the standard deviation.

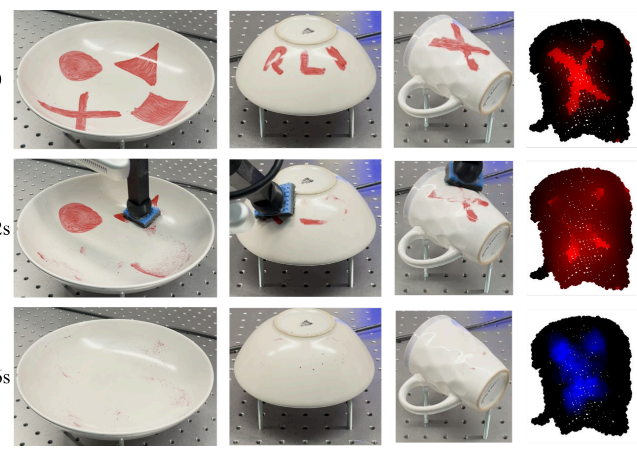

Fig. 10: Real-world experiment of the robot cleaning a plate, a bowl, and a cup. For the first three columns we give snapshots from the initial, intermediate, and final states from top to the bottom. In the last column, we show the target distribution $\boldsymbol{p}$, the simulated potential field $\boldsymbol{u}_t$ and the coverage $\boldsymbol{c}_t$ from top to the bottom.

coverage might be preferable instead of continuing the coverage. To decide when to actually pause and measure the current coverage, we investigated the time evolution of the coverage performance in Figure 9. For the cup and the bunny, we see that the coverage reaches a steady state around the 200-th timestep, while for the plate, this occurs around the 500-th timestep. Still, we can identify the steepest increase in the coverage occurring until the 150-th timestep. Accordingly, we recommend the strategy to pause the coverage at roughly 200 timesteps, measure the actual coverage, and continue the coverage. This would potentially help in the cases where we have unconnected regions (various modes), because discontinuous jumps between the disjoint regions might be quicker and easier than following the surface. All that said, these claims require further testing and experimentation, which are left to be investigated in future work.

C. Force Control

We demonstrated that the proposed method can perform closed-loop tactile ergodic control in the real world with unknown objects and target distributions, as depicted in Figure 10. The primary challenge, however, is to be keeping in contact with the surface without applying excessive force. This is mainly due to the insufficient depth accuracy of the camera, and uncertain dimensions of the mechanical system. A suboptimal solution is to use a compliant controller and adjust the penetration depth of the impedance target. A too compliant controller would, however, reduce the tracking precision and the uncertainty in the penetration depth could lead to unnecessarily high contact forces that might damage the object. More importantly, high contact forces result in high friction that further reduces the reference tracking performance.

Our solution to this problem was to introduce tactile feedback from the wrist-mounted force and torque sensor and closed-loop tracking of a reference contact force. In general, the commands generated by the force controllers conflict with the position controllers and result in competing objectives. We overcome this problem by posing the objective as line tracking instead of position tracking. This forces the agent to be on the line but free to move along the line. Accordingly, the force and the line controller can simultaneously be active without conflicting objectives or rigorous parameter tuning.

D. Comparisons

We compared our method with state-of-the-art approaches in Table I. Since the methods are not comparable in all aspects, we discuss the advantages and disadvantages of our method in three parts: (i) ergodic coverage; (ii) tactile interactions; and (iii) tactile coverage.

1) Ergodic Coverage: In the literature, the only other ergodic coverage method on curved surfaces is the finite element-based HEDAC [46]. This work presents an offline planning method on meshes for visual inspection using multiple aerial vehicles. Accordingly, our method extends the state of the art in ergodic coverage on curved surfaces by being the first formulation (i) working on point clouds, (ii) providing closed-loop coverage using vision, and (iii) performing tactile coverage. Furthermore, as we showed in the experiments in Section V-B1, our approach vastly outscales the finite element-based HEDAC in terms of computation time for the preprocessing step. It is also important to note that, due to the generality of the underlying ergodic control formulation that we are using, our method could be applied to their use-case as well.

2) Tactile Interactions: To ensure contact during tactile exploration, the usage of a unified force-impedance control scheme was proposed [76]. The general idea is similar to ours, in the sense that the controller is required to track a given reference while exerting a force on the surface. The main difference stems from the formulation of the reference for the impedance behavior. While their method tracks a full Cartesian pose, our impedance controller tracks a line. The main difference here is that our method imposes less constraints on the reference tracking, which leaves more degrees of freedom for secondary tasks, such as tracking the force objective. Hence, we require no additional tuning to integrate these objectives, whereas their method uses a passivity-based design to ensure the stability of the combined controller.

3) Tactile Coverage: Concerning the problem of using a manipulator for tactile coverage on curved surfaces, we compare our method to the online sampling-based planner presented in [20]. Unlike the more general point cloud representation that we are using, this method operates on meshes. However, it includes the planning of the configuration of the target object. This is currently a limitation of our approach, since we assume the object to be fixed and consider only a single viewpoint. Although this configuration planner is considered to be independent of the coverage at a given configuration, it could be easily combined with our method. In contrast to our myopic feedback controller, they use trajectory planning, which requires a predefined planning horizon using a number of passes for covering discrete patches. For tactile coverage tasks, this can be extremely challenging to estimate beforehand. Our method does not suffer from this limitation, since ergodicity guarantees revisiting continuous areas according to the target distribution over an infinite time horizon. In addition, their approach is based on generating splines that connect the waypoints. This has two issues: if the points are not densely sampled, there is no guarantee that the resulting spline would be on the surface; and conversely, if the points are densely sampled, then the spline would be very complex and not smooth. Accordingly, this approach would not scale to complex surfaces and target distributions. Our approach, on the other hand, uses a feedback controller to stay in contact with the surface, where the local references are coming from the surface-constrained ergodic controller. Hence, our approach is mainly limited by the robot's geometry with respect to the complexity of the object, which could also be mitigated by changing its configuration online.

VII. Conclusion and Future Work

In this paper, we presented the first closed-loop ergodic coverage method on point clouds to address the tactile coverage tasks on curved surfaces. Tactile coverage tasks are challenging to model due to complex physical interactions. We use vision to jointly capture the surface geometry and the target distribution as a point cloud and directly use this representation as input. Then, we propagate the information regarding the coverage target to our robot using a diffusion process on the point cloud. Here, we use ergodicity to relate the spatial distribution to the number of visits required for coverage in an infinite-horizon formulation. We leverage a spectral formulation to trade-off the accuracy of the diffusion computation with its computational complexity. To find a favorable compromise between the two, we tested the dependency of the coverage performance to the hyperparameters in kinematic simulation experiments. Next, we demonstrated the method in a real-world setting by cleaning previously unknown curved surfaces with arbitrary human-drawn distributions. We observed that our method can indeed adapt and generalize to different objects and distributions on the fly.

In some scenarios, such as surface inspection, sanding, or mechanical palpation, measuring the actual coverage is not straightforward using an RGB-D camera. Still, we can use cleaning as a proxy task such that a human expert can mark the regions that need to be inspected with an easy-to-remove marker. Then, the robot's progress would be detectable by a camera. Accordingly, our method provides an interesting human-robot interaction modality using annotations and markings of an expert for tactile robotics tasks.

As discussed in Section VI-D3, the primary limitation of our work is fixing the object pose during the operation. Therefore, we plan to extend our method to scenarios where the object is grasped by a second manipulator and can be reconfigured for covering regions that otherwise would be unreachable due to either collisions or joint limits. Although this problem is easy to address by sampling discrete configurations, as previously done in [20], our goal is to extend our method to handle this problem in a continuous manner using a control approach.

Another promising extension of our method is automating the collection of visuotactile datasets. In this setting, one can combine our method with a vision-based active learning module such as [61], which estimate high tactile-information regions on the surface. Then, our controller could be used to collect data from these regions with a multi-modal tactile sensor.

References

[1] M. Cakmak and L. Takayama, "Towards a comprehensive chore list for domestic robots," in *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, Tokyo, Japan: IEEE, Mar. 2013, pp. 93–94. DOI: 10.1109/HRI.2013.6483517.

[2] P. Veerajagadheswar, S. Yuyao, P. Kandasamy, M. R. Elara, and A. A. Hayat, "S-Sacrr: A Staircase and Slope Accessing Reconfigurable Cleaning Robot and its Validation," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 4558–4565, Apr. 2022. DOI: 10.1109/LRA.2022.3151572.

[3] Y. Sun, Z. Jing, P. Dong, J. Huang, W. Chen, and H. Leung, "A Switchable Unmanned Aerial Manipulator System for Window-Cleaning Robot Installation," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3483–3490, Apr. 2021. DOI: 10.1109/LRA.2021.3062795.

[4] B. Maric, A. Mutka, and M. Orsag, "Collaborative Human-Robot Framework for Delicate Sanding of Complex Shape Surfaces," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2848–2855, Apr. 2020. DOI: 10.1109/LRA.2020.2969951.

[5] K. Karacan, H. Sadeghian, R. Kirschner, and S. Haddadin, "Passivity-Based Skill Motion Learning in Stiffness-Adaptive Unified Force-Impedance Control," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Kyoto, Japan: IEEE, Oct. 23, 2022, pp. 9604–9611. DOI: 10.1109/IROS47612.2022.9981728.

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

[7] I. F. Onstein, O. Semeniuta, and M. Bjerkeng, "Deburring Using Robot Manipulators: A Review," in *2020 3rd International Symposium on Small-scale Intelligent Manufacturing Systems (SIMS)*, Gjovik, Norway: IEEE, Jun. 2020, pp. 1–7. DOI: 10.1109/SIMS49386.2020.9121490.

[8] P. Pfändler, K. Bodie, G. Crotta, M. Pantic, R. Siegwart, and U. Angst, "Non-destructive corrosion inspection of reinforced concrete structures using an autonomous flying robot," *Automation in Construction*, vol. 158, p. 105 241, Feb. 2024. DOI: 10.1016/j.autcon.2023.105241.

[9] E. Ayvali, R. A. Srivatsan, L. Wang, R. Roy, N. Simaan, and H. Choset, "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)*, May 2016, pp. 931–936. DOI: 10.1109/ICRA.2016.7487225.

[10] Y. Yan and J. Pan, "Fast Localization and Segmentation of Tissue Abnormalities by Autonomous Robotic Palpation," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1707–1714, Apr. 2021. DOI: 10.1109/LRA.2021.3058870.

[11] Z. Jiang, N. Danis, Y. Bi, *et al.*, "Precise Repositioning of Robotic Ultrasound: Improving Registration-based Motion Compensation using Ultrasound Confidence Optimization," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–11, 2022. DOI: 10.1109/TIM.2022.3200360.

[12] J. Fu, I. Burzo, E. Iovene, J. Zhao, G. Ferrigno, and E. De Momi, "Optimization-Based Variable Impedance Control of Robotic Manipulator for Medical Contact Tasks," *IEEE Transactions on Instrumentation and Measurement*, vol. 73, pp. 1–8, 2024. DOI: 10.1109/TIM.2024.3372209.

[13] R. C. Luo, C. P. Tsai, and K. C. Hsieh, "Robot assisted tapping control for therapeutical percussive massage applications," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, Singapore: IEEE, May 2017, pp. 3606–3611. DOI: 10.1109/ICRA.2017.7989415.

[14] M. Khoramshahi, G. Henriks, A. Naef, S. S. M. Salehian, J. Kim, and A. Billard, "Arm-hand motion-force coordination for physical interactions with non-flat surfaces using dynamical systems: Toward compliant robotic massage," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France: IEEE, May 2020, pp. 4724–4730. DOI: 10.1109/ICRA40945.2020.9196593.

[15] Chih-Hung King, T. L. Chen, A. Jain, and C. C. Kemp, "Towards an assistive robot that autonomously performs bed baths for patient hygiene," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei: IEEE, Oct. 2010, pp. 319–324. DOI: 10.1109/IROS.2010.5649101.

[16] R. Madan, S. Valdez, D. Kim, *et al.* "RABBIT: A Robot-Assisted Bed Bathing System with Multimodal Perception and Integrated Compliance." (Jan. 26, 2024), [Online]. Available: http://arxiv.org/abs/2401.15159 (visited on 07/10/2024), pre-published.

[17] T. Lin, Y. Zhang, Q. Li, *et al.* "Learning Visuotactile Skills with Two Multifingered Hands." (May 22, 2024), [Online]. Available: http://arxiv.org/abs/2404.16823 (visited on 07/09/2024), pre-published.

[18] A. Aydinoglu and M. Posa, "Real-Time Multi-Contact Model Predictive Control via ADMM," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, May 2022, pp. 3414–3421. DOI: 10.1109/ICRA46639.2022.9811957.

[19] J. Kim, A. K. Mishra, R. Limosani, *et al.*, "Control strategies for cleaning robots in domestic applications: A comprehensive review," *International Journal of Advanced Robotic Systems*, vol. 16, no. 4, p. 1 729 881 419 857 432, Jul. 1, 2019. DOI: 10.1177/1729881419857432.

[20] A. M. Kabir, K. N. Kaipa, J. Marvel, and S. K. Gupta, "Automated Planning for Robotic Cleaning Using Multiple Setups and Oscillatory Tool Motions," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 3, pp. 1364–1377, Jul. 2017. DOI: 10.1109/TASE.2017.2665460.

[21] K. Mertens, B. De Ketelaere, B. Kamers, *et al.*, "Dirt detection on brown eggs by means of color computer vision," *Poultry Science*, vol. 84, no. 10, pp. 1653–1659, Oct. 2005. DOI: 10.1093/ps/84.10.1653.

[22] D. Canedo, P. Fonseca, P. Georgieva, and A. J. R. Neves, "A Deep Learning-Based Dirt Detection Computer Vision System for Floor-Cleaning Robots with Improved Data Collection," *Technologies*, vol. 9, no. 4, p. 94, Dec. 1, 2021. DOI: 10.3390/technologies9040094.

[23] G. Mathew and I. Mezic, "Spectral Multiscale Coverage: A uniform coverage algorithm for mobile sensor networks," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) Held Jointly with 2009 28th Chinese Control Conference*, Dec. 2009, pp. 7872–7877. DOI: 10.1109/CDC.2009.5400401.

[24] A. Zelinsky, R. A. Jarvis, J. C. Byrne, and S. Yuta, "Planning Paths of Complete Coverage of an Unstructured Environment by a Mobile Robot,"

[25] J.-C. Latombe, *Robot Motion Planning*. Boston, MA: Springer US, 1991. DOI: 10.1007/978-1-4615-4022-9.

[26] H. Choset, "Coverage for robotics – A survey of recent results,"

[27] M. Ollis and A. Stentz, "First results in vision-based crop line tracking," in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 1, Minneapolis, MN, USA: IEEE, 1996, pp. 951–956. DOI: 10.1109/ROBOT.1996.503895.

[28] H. Choset and P. Pignon, "Coverage Path Planning: The Boustrophedon Cellular Decomposition," in *Field and Service Robotics*, A. Zelinsky, Ed., London: Springer London, 1998, pp. 203–209. DOI: 10.1007/978-1-4471-1273-0_32.

[29] E. U. Acar, H. Choset, A. A. Rizzi, P. N. Atkar, and D. Hull, "Morse Decompositions for Coverage Tasks," *The International Journal of Robotics Research*, vol. 21, no. 4, pp. 331–344, Apr. 2002. DOI: 10.1177/027836402320556359.

[30] S. Hert and S. Tiwari, "A terrain-covering algorithm for an AUV,"

[31] P. Atkar, D. Conner, A. Greenfield, H. Choset, and A. Rizzi, "Hierarchical Segmentation of Piecewise Pseudoextruded Surfaces for Uniform Coverage," *IEEE Transactions on Automation Science and Engineering*, vol. 6, no. 1, pp. 107–120, Jan. 2009. DOI: 10.1109/TASE.2008.916768.

[32] H. Zhu, N. R. J. Lawrance, R. Siegwart, and J. Alonso-Mora. "Online Informative Path Planning for Active Information Gathering of a 3D Surface." (Mar. 17, 2021), pre-published.

[33] C. S. Tan, R. Mohd-Mokhtar, and M. R. Arshad, "A Comprehensive Review of Coverage Path Planning in Robotics Using Classical and Heuristic Algorithms," *IEEE Access*, vol. 9, pp. 119 310–119 342, 2021. DOI: 10.1109/ACCESS.2021.3108177.

[34] V.-T. Do and Q.-C. Pham, "Geometry-Aware Coverage Path Planning for Depowdering on Complex 3D Surfaces," *IEEE Robotics and Automation Letters*, vol. 8, no. 9, pp. 5552–5559, Sep. 2023. DOI: 10.1109/LRA.2023.3296943.

[35] G. Mathew and I. Mezić, "Metrics for ergodicity and design of ergodic dynamics for multi-agent systems," *Physica D: Nonlinear Phenomena*, vol. 240, no. 4-5, pp. 432–442, Feb. 2011. DOI: 10.1016/j.physd.2010.10.010.

[36] T. A. Berrueta, A. Pinosky, and T. D. Murphey. "Maximum Diffusion Reinforcement Learning." (Sep. 28, 2023), [Online]. Available: http://arxiv.org/abs/2309.15293 (visited on 10/02/2023), pre-published.

[37] S. Shetty, J. Silvério, and S. Calinon, "Ergodic Exploration Using Tensor Train: Applications in Insertion Tasks," *IEEE Transactions on Robotics*, vol. 38, no. 2, pp. 906–921, 2 Apr. 2022. DOI: 10.1109/TRO.2021.3087317.

[38] C. Lerch, D. Dong, and I. Abraham, "Safety-Critical Ergodic Exploration in Cluttered Environments via Control Barrier Functions," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, May 2023, pp. 10 205–10 211. DOI: 10.1109/ICRA48891.2023.10161032.

[39] D. Dong, H. Berger, and I. Abraham, "Time Optimal Ergodic Search," in *Robotics: Science and Systems XIX*, Robotics: Science and Systems Foundation, Jul. 10, 2023. DOI: 10.15607/RSS.2023.XIX.082.

[40] A. Seewald, C. J. Lerch, M. Chancán, A. M. Dollar, and I. Abraham. "Energy-Aware Ergodic Search: Continuous Exploration for Multi-Agent Systems with Battery Constraints." (Oct. 13, 2023), [Online]. Available: http://arxiv.org/abs/2310.09470 (visited on 10/24/2023), pre-published.

[41] H. Jacobs, S. Nair, and J. Marsden, "Multiscale surveillance of Riemannian manifolds," in *Proceedings of the 2010 American Control Conference*, Jun. 2010, pp. 5732–5737. DOI: 10.1109/ACC.2010.5531152.

[42] M. Sun, A. Gaggar, P. Trautman, and T. Murphey. "Fast Ergodic Search with Kernel Functions." (Mar. 3, 2024), [Online]. Available: http://arxiv.org/abs/2403.01536 (visited on 03/10/2024), pre-published.

[43] S. Ivić, B. Crnković, and I. Mezić, "Ergodicity-Based Cooperative Multiagent Area Coverage via a Potential Field," *IEEE Transactions on Cybernetics*, vol. 47, no. 8, pp. 1983–1993, Aug. 2017. DOI: 10.1109/TCYB.2016.2634400.

[44] S. Ivić, A. Sikirica, and B. Crnković, "Constrained multi-agent ergodic area surveying control based on finite element approximation of the potential field," *Engineering Applications of Artificial Intelligence*, vol. 116, p. 105 441, Nov. 2022. DOI: 10.1016/j.engappai.2022.105441.

[45] B. Crnković, S. Ivić, and M. Zovko. "Fast algorithm for centralized multi-agent maze exploration." (Oct. 3, 2023), [Online]. Available: http://arxiv.org/abs/2310.02121 (visited on 02/01/2024), pre-published.

[46] S. Ivić, B. Crnković, L. Grbčić, and L. Matleković, "Multi-UAV trajectory planning for 3D visual inspection of complex structures," *Automation in Construction*, vol. 147, p. 104 709, Mar. 1, 2023. DOI: 10.1016/j.autcon.2022.104709.

[47] N. Sharp, S. Attaiki, K. Crane, and M. Ovsjanikov. "DiffusionNet: Discretization Agnostic Learning on Surfaces." (Jan. 7, 2022), pre-published.

[48] K. Crane, F. De Goes, M. Desbrun, and P. Schröder, "Digital geometry processing with discrete exterior calculus," in *ACM SIGGRAPH 2013*

Courses, Anaheim California: ACM, Jul. 21, 2013, pp. 1–126. DOI: 10.1145/2504435.2504442.

[49] M. Belkin, J. Sun, and Y. Wang, "Constructing Laplace Operator from Point Clouds in R^d," in *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, Society for Industrial and Applied Mathematics, Jan. 4, 2009, pp. 1031–1040. DOI: 10.1137/1.9781611973068.112.

[50] Y. Liu, B. Prabhakaran, and X. Guo, "Point-Based Manifold Harmonics," *IEEE Trans. Visual. Comput. Graphics*, vol. 18, no. 10, pp. 1693–1703, 10 Oct. 2012. DOI: 10.1109/TVCG.2011.152.

[51] J. Cao, A. Tagliasacchi, M. Olson, H. Zhang, and Z. Su, "Point Cloud Skeletons via Laplacian Based Contraction," in *2010 Shape Modeling International Conference*, Aix-en-Provence, France: IEEE, Jun. 2010, pp. 187–197. DOI: 10.1109/SMI.2010.25.

[52] N. Sharp and K. Crane, "A Laplacian for Nonmanifold Triangle Meshes," *Computer Graphics Forum*, vol. 39, no. 5, pp. 69–80, 5 2020. DOI: 10.1111/cgf.14069.

[53] E. Acar, H. Choset, and Ji Yeong Lee, "Sensor-based coverage with extended range detectors," *IEEE Transactions on Robotics*, vol. 22, no. 1, pp. 189–198, Feb. 2006. DOI: 10.1109/TRO.2005.861455.

[54] V. Shivashankar, R. Jain, U. Kuter, and D. Nau, "Real-Time Planning for Covering an Initially-Unknown Spatial Environment,"

[55] I. Abraham, A. Prabhakar, M. J. Z. Hartmann, and T. D. Murphey, "Ergodic Exploration using Binary Sensing for Non-Parametric Shape Estimation," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 827–834, Apr. 2017. DOI: 10.1109/LRA.2017.2654542.

[56] C. Bilaloglu, T. Löw, and S. Calinon, "Whole-Body Ergodic Exploration with a Manipulator Using Diffusion," *IEEE Robot. Autom. Lett.*, pp. 1–7, 2023. DOI: 10.1109/LRA.2023.3329755.

[57] S. Chitta, J. Sturm, M. Piccoli, and W. Burgard, "Tactile Sensing for Mobile Manipulation," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 558–568, Jun. 2011. DOI: 10.1109/TRO.2011.2134130.

[58] E. Ayvali, R. A. Srivatsan, L. Wang, R. Roy, N. Simaan, and H. Choset, "Using Bayesian optimization to guide probing of a flexible environment for simultaneous registration and stiffness mapping," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 931–936. DOI: 10.1109/ICRA.2016.7487225.

[59] P. Chalasani, L. Wang, R. Roy, N. Simaan, R. H. Taylor, and M. Kobilarov, "Concurrent nonparametric estimation of organ geometry and tissue stiffness using continuous adaptive palpation," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 4164–4171. DOI: 10.1109/ICRA.2016.7487609.

[60] H. Salman, E. Ayvali, R. A. Srivatsan, *et al.*, "Trajectory-Optimized Sensing for Active Search of Tissue Abnormalities in Robotic Surgery," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, QLD: IEEE, May 2018, pp. 5356–5363. DOI: 10.1109/ICRA.2018.8460936.

[61] J. Ketchum, A. Prabhakar, and T. D. Murphey. "Active Exploration for Real-Time Haptic Training." (May 20, 2024), [Online]. Available: http://arxiv.org/abs/2405.11776 (visited on 07/08/2024), pre-published.

[62] M. Neidhardt, R. Mieling, M. Bengs, and A. Schlaefer, "Optical force estimation for interactions between tool and soft tissues," *Sci Rep*, vol. 13, no. 1, p. 506, 1 Jan. 10, 2023. DOI: 10.1038/s41598-022-27036-7.

[63] E. Vignali, E. Gasparotti, K. Capellini, *et al.*, "Modeling biomechanical interaction between soft tissue and soft robotic instruments: Importance of constitutive anisotropic hyperelastic formulations," *The International Journal of Robotics Research*, vol. 40, no. 1, pp. 224–235, Jan. 2021. DOI: 10.1177/0278364920927476.

[64] W. X. Ng, H. K. Chan, W. K. Teo, and I.-M. Chen, "Programming a Robot for Conformance Grinding of Complex Shapes by Capturing the Tacit Knowledge of a Skilled Operator," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 2, pp. 1020–1030, Apr. 2017. DOI: 10.1109/TASE.2015.2474708.

[65] U. Martinez-Hernandez, T. J. Dodd, M. H. Evans, T. J. Prescott, and N. F. Lepora, "Active sensorimotor control for tactile exploration," *Robotics and Autonomous Systems*, vol. 87, pp. 15–27, Jan. 2017. DOI: 10.1016/j.robot.2016.09.014.

[66] T. Lew, S. Singh, M. Prats, *et al.*, "Robotic Table Wiping via Reinforcement Learning and Whole-body Trajectory Optimization," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, May 2023, pp. 7184–7190. DOI: 10.1109/ICRA48891.2023.10161283.

[67] N. Saito, D. Wang, T. Ogata, H. Mori, and S. Sugano, "Wiping 3D-objects using Deep Learning Model based on Image/Force/Joint Information," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA: IEEE, Oct. 24, 2020, pp. 10 152–10 157. DOI: 10.1109/IROS45743.2020.9341275.

[68] A. Kalinowska, A. Prabhakar, K. Fitzsimons, and T. Murphey, "Ergodic imitation: Learning from what to do and what not to do," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, May 2021, pp. 3648–3654. DOI: 10.1109/ICRA48506.2021.9561746.

[69] T. Löw, J. Maceiras, and S. Calinon, "drozBot: Using Ergodic Control to Draw Portraits," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 728–11 734, 4 Oct. 2022. DOI: 10.1109/LRA.2022.3186735.

[70] D. Hestenes, "New Tools for Computational Geometry and Rejuvenation of Screw Theory," in *Geometric Algebra Computing: In Engineering and Computer Science*, E. Bayro-Corrochano and G. Scheuermann, Eds., London: Springer, 2010, pp. 3–33. DOI: 10.1007/978-1-84996-108-0_1.

[71] M. Caron, H. Touvron, I. Misra, *et al.* "Emerging Properties in Self-Supervised Vision Transformers." (May 24, 2021), [Online]. Available: http://arxiv.org/abs/2104.14294 (visited on 07/06/2024), pre-published.

[72] D. Hildenbrand, *Foundations of Geometric Algebra Computing* (Geometry and Computing). Berlin, Heidelberg: Springer, 2013, vol. 8. DOI: 10.1007/978-3-642-31794-1.

[73] A. Nealen and T. U. Darmstadt, "An As-Short-As-Possible Introduction to the Least Squares, Weighted Least Squares and Moving Least Squares Methods for Scattered Data Approximation and Interpolation,"

[74] J. Lasenby, H. Hadfield, and A. Lasenby, "Calculating the Rotor Between Conformal Objects," *Adv. Appl. Clifford Algebras*, vol. 29, no. 5, p. 102, Oct. 22, 2019. DOI: 10.1007/s00006-019-1014-8.

[75] T. Löw and S. Calinon, "Geometric Algebra for Optimal Control With Applications in Manipulation Tasks," *IEEE Transactions on Robotics*, vol. 39, no. 5, pp. 3586–3600, 2023. DOI: 10.1109/TRO.2023.3277282.

[76] K. Karacan, D. Grover, H. Sadeghian, F. Wu, and S. Haddadin, "Tactile Exploration Using Unified Force-Impedance Control," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 5015–5020, 2023. DOI: 10.1016/j.ifacol.2023.10.1279.