ProSIP: Probabilistic Surface Interaction Primitives for Learning of Robotic Cleaning of Edges

Christoph Unger[1], Christian Hartl-Nesic[1], Minh Nhat Vu[1,2], and Andreas Kugi[1,2]

***Abstract*— Learning from demonstration (LfD) has emerged as a promising approach enabling robots to acquire complex tasks directly from human demonstrations. However, tasks involving surface interactions on freeform 3D surfaces present unique challenges in modeling and execution, especially when geometric variations exist between demonstrations and robot execution. This paper proposes a novel framework called *probabilistic surface interaction primitives* (ProSIP), which systematically incorporates the surface path and the local surface features into the learning procedure. An instrumented tool allows seamless recording and execution of human demonstrations. By design, ProSIPs are independent of time, invariant to rigid-body displacements, and apply to any robotic platform with a Cartesian controller. The framework is employed for an edge-cleaning task of bathroom sinks. The generalization capability to various object geometries and significantly distorted objects is demonstrated. Simulations and an experimental setup with a 9-degrees-of-freedom robotic platform confirm the performance.**

I. INTRODUCTION

Robotic systems are employed to perform numerous different tasks in classical automation. Recently, robots have become more available in emerging areas where they were rarely found before, e.g., in industrial high-mix/low-volume production [1], [2], craftsmanship [3], households [4], and elderly care [5]. In these areas, the tasks to be performed are challenging due to the high degree of individualization. At the same time, robotic systems need to be programmed without expert robot knowledge. One promising approach to address these challenges is *learning from demonstration* (LfD), a methodology that enables robots to learn and adapt complex tasks directly from human demonstrations. LfD has been applied to a variety of tasks ranging from assembly [2], [6] to dynamically catching a water bottle [7], [8].

Frequent tasks in robotic manufacturing are polishing [9], [10], sanding [11], and cleaning [12] planar 3D surfaces, e.g., cleaning ceramic surfaces in the bathroom or even personal hygiene. In an LfD framework, these tasks are particularly challenging to model if the geometric shape of the surface differs between the demonstration and the robotic execution. Additionally, the temporal alignment of multiple demonstrations poses another challenge [13], [14] since the demonstrations may vary in geometric length, instantaneous execution speed, and total duration. In order to capture the demonstrated surface process appropriately, the geometry of the underlying surface and the surface path have to be systematically considered during learning.

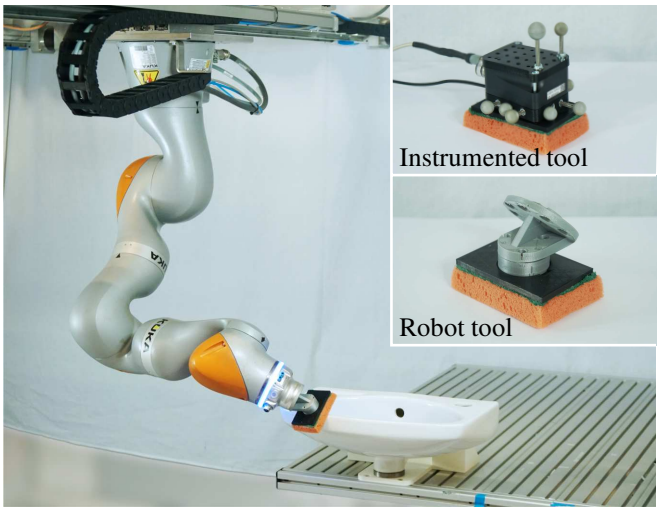

Fig. 1: Experimental setup for the edge-cleaning task on bathroom sinks. The demonstration data is recorded using an instrumented tool. The trajectories generated by the ProSIP framework are executed on a 9-DoF robot system. The robot cleans dirt spots from different edges of different object geometries.

This work proposes an LfD framework for an edge-cleaning task of bathroom sinks. In this framework, the main contribution is the *probabilistic surface interaction primitives* (ProSIP) approach, which provides key features for learning processes on freeform 3D surfaces, i.e.,

1) systematically incorporating the underlying surface geometry into the learning procedure,
2) becoming independent of time by considering the surface path and the local surface features and
3) describing the tool motion via a projection of the tool center point (TCP) onto the surface, which makes the framework independent from the employed robotic platform.

The human demonstrations are recorded using a so-called *instrumented tool*, which is a standard tool equipped with additional sensors [15]. Hence, an instrumented tool represents a demonstration interface with which the *record mapping* (for recording the dataset) and the *embodiment mapping* (for execution of learned trajectories) become easier [16], [17]. The execution of the edge-cleaning process is performed using a robotic system with 9 degrees of freedom (DoF) comprising the 7-DoF collaborative robot KUKA LBR iiwa 14 R820 and two orthogonal linear axes to move the robot

[1] C. Unger, C. Hartl-Nesic, M. N. Vu, and A. Kugi are with the Automation & Control Institute (ACIN), TU Wien, Austria (e-mail: {unger,hartl,vu,kugi}@acin.tuwien.ac.at)

[2] M. N. Vu and A. Kugi are with the AIT Austrian Institute of Technology GmbH, Austria (e-mail: {minh.vu,andreas.kugi}@ait.ac.at)

The first and second author have contributed equally to this paper.

base, see Fig. 1. A video of the demonstration is shown in www.acin.tuwien.ac.at/a79d.

II. Related Work

Probabilistic motion primitives (ProMPs) [18] are pioneering approaches to learn from human demonstrations and generate desired trajectories or motor skills in various robotic scenarios using probabilistic models. While ProMPs are formulated for time-driven trajectories, several extensions have been proposed to apply this concept to other scenarios. For example, learning from only a few examples has been addressed in [19], and learning multiple skills by incorporating task descriptions is proposed in [20]. Conditional neural movement primitives (CNMPs) [21] consider high-dimensional inputs using encoder-decoder models for context-based tasks. The paper [22] proposes a Riemannian formulation for ProMPs to enable encoding and retrieving of quaternion trajectories, making it a suitable model for learning complex robot motions in task space. However, the time information contained in the demonstrations needs to be handled by complex temporal and spatial alignment algorithms, e.g., using dynamic time warping (DTW) [23] or rhythmic motion modulation [24], [25]. The framework [26] focuses on rapidly sequencing manipulation skills learned from demonstration, mainly for industrial assembly tasks. The goal is to reduce manual modeling efforts while ensuring flexibility and usability of learned skills in a manufacturing environment. For complex interaction tasks on freeform 3D surfaces, extensions are needed to adapt to the underlying surface geometry.

Learning of non-temporal primitives is proposed in [27] with so-called kernelized motion primitives (KMPs), which can also apply to local (relative) movements and force-based trajectory adaption. The KMPs embed data points to create a reference database involving reference input and latent space variables. Generating a reference trajectory for new observations relies on the entire reference database, similar to Gaussian process regression (GPR). Conditioning new trajectories on the reference database requires significant computational and memory effort. The method proposed in [28] deals with non-temporal constraint primitives for trajectory generation based on geometric constraints. A motion planner based on Monte Carlo tree search and Bayesian optimization is employed to compute feasible constraints in pick-and-place scenarios. Another approach, named arc-length probabilistic movement primitives (AL-ProMP) [29], combines the learning of a normal-force distribution and Cartesian dynamical movement primitives (DMPs) for arc-length motion trajectories given on the surface of an object. The nonlinear force term of the DMPs is learned as a ProMP, while the spatial component is projected onto the surface. In [30], [31], merely point-wise interaction between a robot and an object or human using ProMPs has been introduced for specific sequential interaction goals in the task space. Interaction with partially unknown surfaces is shown in [32]. The proposed framework combines adaptive optimal admittance control with DMP trajectory generation and demonstrates a wiping task on a balloon surface with unknown parameters.

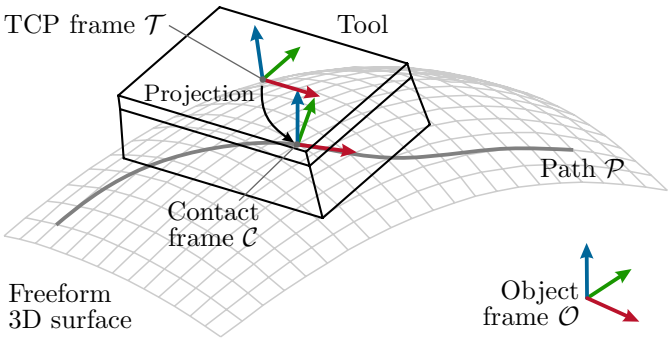

Fig. 2: Schematic illustration of a complex interaction task on a freeform 3D surface.

The above methods can generate new motions similar to the demonstrations by sampling the learned representation. However, the user must explicitly set via-points or use optimization-based algorithms to obtain via-points for sampling, and a progress modulation has to be defined during conditioning. This property limits the ability to generalize motion primitives concerning an underlying surface. The proposed ProSIPs extend the concept of ProMPs to continuous interaction tasks with a tool, described in Cartesian space, on freeform 3D surfaces, which is not possible in state-of-the-art approaches [27], [31]. This is achieved by relating the tool motion via a projection to the surface and incorporating local surface features, e.g., curvatures, into the learning procedure, see Fig. 2. In this way, a probabilistic model is learned by capturing the appropriate time-independent tool motions and uncertainties relative to the projected contact frame. The Cartesian description of the tool motion leads to a platform-independent description of the process in the context of the underlying surface. Finally, new tool motions on objects with similar geometry are generated by inference on a new contact point path on the surface.

III. Problem Statement

A complex interaction task on a freeform 3D surface, e.g., polishing, sanding, and cleaning, is described by the motion of the tool's TCP, the contact point at which the interaction occurs, the path $\mathcal{P}$ along which the contact point moves, and the underlying surface. An interaction task is assumed to strongly depend on the shape of the underlying surface, i.e., flat areas need to be treated differently than sharp edges. The geometric relations are schematically illustrated in Fig. 2. The problem to be solved is developing an LfD framework that allows non-robot experts to teach such tasks to robotic systems. The framework should be formulated generally so that additional signals, e.g., interaction forces and torques, can be incorporated.

In this framework, a few human demonstrations of the interaction task are recorded using an instrumented tool, considered an intuitive interface for humans. The freeform 3D surface is assumed to be known from CAD data or as a reconstruction. The demonstrations may differ in geometric shape and length of the surface path, duration,

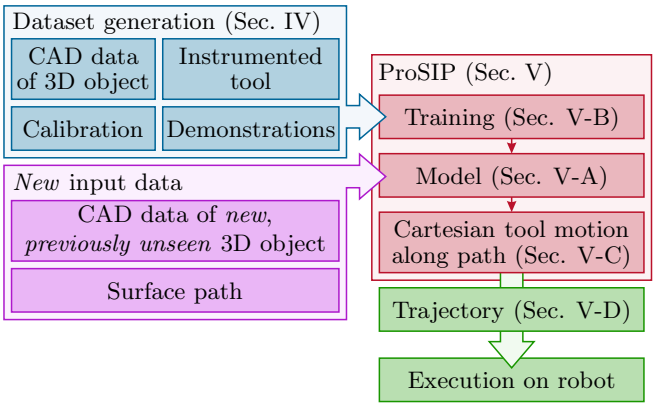

Fig. 3: Overview of the ProSIP framework and dataflow.

and execution speed. The additional sensors integrated into the instrumented tools generate high-quality demonstration data, which allows learning of very complex, precise, and delicate surface processes. In contrast, motion tracking of the human body or hands is insufficient for most processes. The demonstrations are associated with the surface by projection of the TCP, which generates the surface path $\mathcal{P}$ and local contact frames $\mathcal{C}$ for each demonstration, see Fig. 2. Next, ProSIPs are learned from the relative poses (i.e., position and orientation) of the TCP frame $\mathcal{T}$ and the corresponding contact frames $\mathcal{C}$, while incorporating the local surface features of the underlying surface, e.g., the curvature. The ProSIPs are then used to generate tool motions on new, previously unseen objects with similar geometry. This is achieved by inference on a new surface path on the new object, which yields the desired TCP motion. This new surface path may be generated by manual annotation or an automatic (coverage) algorithm [33]. Standard or advanced task-space trajectory planners [34] or robot controllers [35] are then applied to execute the computed motion on the robot and perform the interaction task on the surface. An overview of the framework and the dataflow is depicted in Fig. 3.

This work considers cleaning the edges of bathroom sinks as a complex interaction task, see Fig. 1. A household sponge equipped with tracking markers for an optical tracking system serves as an instrumented tool. The bathroom sinks are reconstructed with high resolution. Cleaning the front edge of the sink is demonstrated, from which the ProSIPs for the edge-cleaning task are learned. A suitable projection algorithm is introduced to account for the fact that the sponge has a flat interaction region. The learned ProSIPs allow us to generate the new tool motions based on a surface path on the edges of a new, previously unseen bathroom sink. Robustness is shown by executing cleaning motions in simulation on objects with different geometry, including significantly distorted bathroom sinks. The proposed approach is evaluated experimentally by cleaning dirt spots on different bathroom sinks on a 9-DoF robotic system.

IV. Edge-Cleaning Dataset for Bathroom Sinks

The dataset for learning ProSIPs for the edge-cleaning task is generated from the recorded measurement signals, i.e., the human demonstrations with the instrumented tool, and

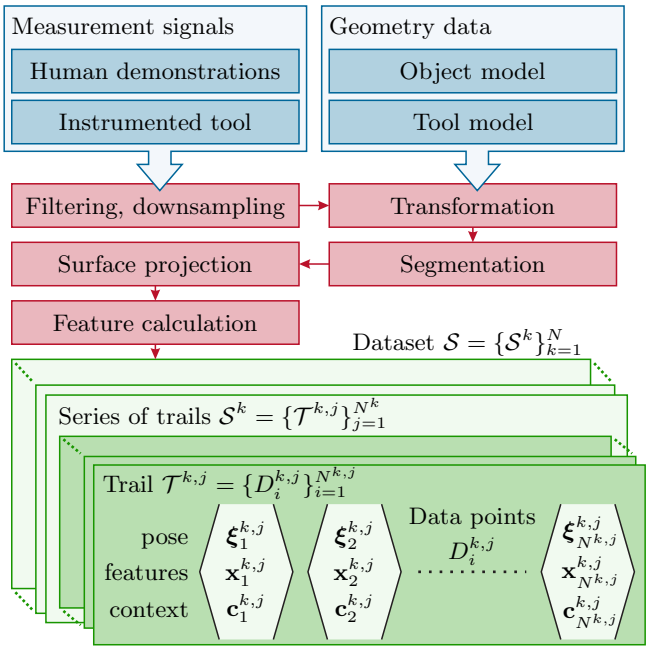

Fig. 4: Scheme for the measurement signals, geometry data, and dataset generation.

geometrical data, i.e., the object and tool model, see Fig. 4. In this section, the necessary steps for the generation of the dataset are explained first (red boxes in Fig. 4), and then the resulting dataset structure is described in detail. Note that this section is tailored to the edge-cleaning task for bathroom sinks but can be adapted straight forward to different tasks and tools. Additionally, further measurement signals, e.g., force/torque (F/T) sensors and IMUs, may be considered in the dataset.

A. Dataset Generation

1) Filtering, downsampling: The measurement signals are filtered using a third-order Butterworth filter to reduce noise. Additionally, the number of samples is reduced appropriately by performing filter-based downsampling.

2) Transformation: Next, the tool motion is transformed into the object frame $\mathcal{O}$, see Fig. 2, using the calibration data, which references the coordinate frame of the optical tracking system with the object frame.

3) Segmentation: The tool motion of each demonstration comprises two phases, i.e., a free-moving phase and an interaction phase with the object. The distinction between those phases is made by computing the closest plane-to-plane distance between the planar surface of the sponge and the freeform 3D surface. To improve robustness, a small neighborhood is considered on the surface at each time step instead of a single point.

4) Projection: Next, the tool motion during the interaction phase is projected onto the surface according to the closest plane-to-plane distance. At each time step, a local contact frame $\mathcal{C}$ on the freeform 3D surface is computed. The orientation of the contact frame is derived from the surface normal vector and the tangent of the surface path. Finally, smoothing is applied to the resulting surface path.

Reprojection guarantees that the surface path lies entirely inside the freeform 3D surface.

5) Feature calculation: In this step, the pose of the TCP frame $\mathcal{T}$ w.r.t. the contact frame $\mathcal{C}$ is computed. Additionally, normalized local surface features, i.e., the principal curvatures, are computed for each time step and are annotated with the tool motion. The normalization of the surface features is based on the geometry of the entire object model and yields an object-independent feature vector. The rotation angle between the path tangent and the direction of the principal curvature is calculated as a context variable.

B. Dataset Structure

The dataset generation yields the complete dataset $\mathcal{S} = \{\mathcal{S}^k\}_{k=1}^{N}$, which consists of N series $\mathcal{S}^k$ of human demonstrations, see Fig. 4. Each series demonstrates a certain aspect or variant of a task. In the context of the edge-cleaning task, a series demonstrates the cleaning of an edge in a specific way, like from left to right or up and down. A series $\mathcal{S}^k = \{\mathcal{T}^{k,j}\}_{j=1}^{N^k}$, $k = 1, \ldots, N$, comprises N^k trials $\mathcal{T}^{k,j}$. Finally, a trial contains a list of observation data points $\mathcal{T}^{k,j} = \{D_i^{k,j}\}_{i=1}^{N^{k,j}}$, where each data point $D_i^{k,j} = \left\langle \boldsymbol{\xi}_i^{k,j}, \mathbf{x}_i^{k,j}, \mathbf{c}_i^{k,j} \right\rangle$ contains the following elements:

- The vector $\left(\boldsymbol{\xi}_i^{k,j}\right)^{\mathrm{T}} = \left[\left(\mathbf{d}_i^{k,j}\right)^{\mathrm{T}} \quad \left(\mathbf{q}_i^{k,j}\right)^{\mathrm{T}}\right]$ describes the pose of the TCP frame $\mathcal{T}$ w.r.t. the contact frame $\mathcal{C}$ with the relative position $\mathbf{d}_i^{k,j} \in \mathbb{R}^3$ and the relative orientation $\mathbf{q}_i^{k,j} \in \mathbb{R}^3$ given as vector part of the corresponding quaternion.
- The normalized local surface feature vector $(\mathbf{x}_i^{k,j})^{\mathrm{T}} = [\kappa_{i,1}^{k,j}, \kappa_{i,2}^{k,j}]$ contains the normalized local mesh principal curvature values calculated at the contact frame. Normalization is done based on the minimum and maximum principal curvature values of the 3D object.
- The context vector $\mathbf{c}_i^{k,j}$ contains information about the local surface properties and the interaction between the tool and the surface. For the edge-cleaning task, the scalar angle $\varphi_i^{k,j}$ between the path tangent and the maximum direction of the principal curvature is used.

V. ProSIP: Model Architecture, Training, and Generation

ProSIPs are novel probabilistic primitives for surface interaction tasks. By systematically incorporating the surface feature vector $\mathbf{x}_i^{k,j}$ and the context vector $\mathbf{c}_i^{k,j}$ of the dataset $\mathcal{S}$, a non-temporal primitive is learned, which generalizes to new, previously unseen surfaces with similar geometry.

First, the model architecture is explained, then the training procedure is briefly described. Finally, the generation of new tool motions and the respective trajectories is detailed.

A. Model Architecture

The ProSIP comprises N linear Gaussian sub-models given by

$$\mathbf{y}_i^{k,j} = \boldsymbol{\Phi}^k(\mathbf{x}_i^{k,j})\mathbf{z}^{k,j} + \boldsymbol{\epsilon}_i^{k,j} \ , \tag{1}$$

where each sub-model k encodes a series $\mathcal{S}^k$, $k = 1, \ldots, N$, of human demonstrations, in which each of the $j = 1, \ldots, N^k$ trials comprises $i = 1, \ldots, N^{k,j}$ data points, see Section IV. In (1), $\mathbf{y}_i^{k,j} \in \mathbb{R}^{N_\mathrm{y}}$ is the observation vector, $\mathbf{z}^{k,j} \in \mathbb{R}^{N_\mathrm{z}}$ is the latent variable, and $\boldsymbol{\epsilon}_i^{k,j} \sim \mathcal{N}(\mathbf{0}, \epsilon\mathbf{I})$, $\epsilon > 0$, is the entropy. The latent variable $\mathbf{z}^{k,j}$ is partitioned into the vectors $\mathbf{z}_d^{k,j} \in \mathbb{R}^{N_{\mathrm{z},d}}$, $d = 1, \ldots, N_\mathrm{y}$, and the total size of the latent space is $N_\mathrm{z} = \sum_{d=1}^{N_\mathrm{y}} N_{\mathrm{z},d}$. Similarly, the surface feature vector $\mathbf{x}_i^{k,j}$ is partitioned into sub-feature vectors $\mathbf{x}_{i,h}^{k,j}$, $h = 1, \ldots, N_\mathrm{x}$.

While the employed structure is similar to the classical ProMPs [19], the *key difference* is that the kernel matrix $\boldsymbol{\Phi}^k(\mathbf{x}_i^{k,j})$ is a function of the surface feature vector $\mathbf{x}_i^{k,j}$ only and is independent of time or progress. This allows to learn the appropriate tool pose relative to the surface $\boldsymbol{\xi}_i^{k,j}$ in the corresponding context $\mathbf{c}_i^{k,j}$ only in terms of the surface features $\mathbf{x}_i^{k,j}$. Both vectors are contained in the observation vector $\mathbf{y}_i^{k,j} = \left[\boldsymbol{\xi}_i^{k,j}, \mathbf{c}_i^{k,j}\right]^{\mathrm{T}}$.

The kernel matrix is given by

$$\boldsymbol{\Phi}^k(\mathbf{x}_i^{k,j}) = \begin{bmatrix} \boldsymbol{\phi}_1^k(\mathbf{x}_i^{k,j}) & \ldots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \ldots & \boldsymbol{\phi}_{N_\mathrm{y}}^k(\mathbf{x}_i^{k,j}) \end{bmatrix} \tag{2}$$

$$\boldsymbol{\phi}_d^k(\mathbf{x}_i^{k,j}) = \bigotimes_{h=1}^{N_\mathrm{x}} \boldsymbol{\psi}_{d,h}^k(\mathbf{x}_{i,h}^{k,j}), \quad d = 1, \ldots, N_\mathrm{y} \ . \tag{3}$$

The elements of the vector basis functions $\boldsymbol{\psi}_{d,h}^k(\mathbf{x}_{i,h}^{k,j})$ are chosen as radial basis functions (RBF) [36]

$$\psi_{d,h,g}^k(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{r}_{d,h,g}^k\|^2}{2(\sigma_{d,h,g}^k)^2}\right) \ , \quad g = 1, \ldots, N_{\mathrm{z},d} \ , \tag{4}$$

with the centers $\mathbf{r}_{d,h,g}^k$ and the widths $\sigma_{d,h,g}^k$. Hence, each sub-feature vector $\mathbf{x}_{i,h}^{k,j}$ has its own proper kernel $\boldsymbol{\psi}_{d,h}^k$, which are combined using the aggregation with the element-wise multiplication $\otimes$ [36]. Note that each vector $\mathbf{z}_d^{k,j}$ is associated with a single element $y_{i,d}^{k,j}$ of the observation vector $\mathbf{y}_i^{k,j}$, $d = 1, \ldots, N_\mathrm{y}$, see (1) and (2).

Remark 1: The $k = 1, \ldots, N$ Gaussian sub-models in (1) may differ from each other in terms of the dimension of the latent space N_z and its partitioning $N_{\mathrm{z},d}$, $d = 1, \ldots, N_\mathrm{y}$, the dimension of the feature vector N_x and its partitioning $N_{\mathrm{x},h}$, $h = 1, \ldots, N_\mathrm{x}$, and also the size of the observation vector N_y. Additional design choices are the aggregation operator $\otimes$ and the basis functions (4). The observation vector $\mathbf{y}_i^{k,j}$ of each model k must describe the pose of the tool relative to the surface.

The encoding distribution $p(\mathbf{z}^{k,j}|\boldsymbol{\theta}_\mathrm{z}^k)$ of each sub-model (1) is assumed to be a Gaussian distribution of the form

$$p(\mathbf{z}^{k,j}|\boldsymbol{\theta}_\mathrm{z}^k) = \\ = \mathcal{N}\left(\begin{bmatrix} \mathbf{z}_1^{k,j} \\ \vdots \\ \mathbf{z}_{N_\mathrm{y}}^{k,j} \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\mu}_{\mathrm{z},1}^k \\ \vdots \\ \boldsymbol{\mu}_{\mathrm{z},N_\mathrm{y}}^k \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{\mathrm{z},1,1}^k & \ldots & \boldsymbol{\Sigma}_{\mathrm{z},1,N_\mathrm{y}}^k \\ \vdots & \ldots & \vdots \\ \boldsymbol{\Sigma}_{\mathrm{z},N_\mathrm{y},1}^k & \ldots & \boldsymbol{\Sigma}_{\mathrm{z},N_\mathrm{y},N_\mathrm{y}}^k \end{bmatrix}\right),$$

and is learned from a single demonstration series $\mathcal{S}^k$. The parameters $\boldsymbol{\theta}_\mathrm{z}^k = \{\boldsymbol{\mu}_\mathrm{z}^k, \boldsymbol{\Sigma}_\mathrm{z}^k\}$ comprise the means $\boldsymbol{\mu}_{\mathrm{z},d}^k$, $d = 1, \ldots, N_\mathrm{y}$, and the covariance matrices $\boldsymbol{\Sigma}_{\mathrm{z},a,b}^k \in \mathbb{R}^{N_{\mathrm{z},a} \times N_{\mathrm{z},b}}$,

$a, b \in \{1, \ldots, N_\mathrm{y}\}$. The probability distribution of the latent space variable $\mathbf{z}^k$ for the series $\mathcal{S}^k$ is given by the joint distribution $p(\mathbf{z}^k)$ of all independent and identically distributed (i.i.d.) variables $\mathbf{z}^{k,j}$, reading as

$$p(\mathbf{z}^k) = p(\mathbf{z}^{k,1}, \ldots, \mathbf{z}^{k,N^k}) = \prod_{j=1}^{N^k} p(\mathbf{z}^{k,j}|\boldsymbol{\theta}_\mathrm{z}^k) \ . \quad (5)$$

Analogous to [19], a regularization in form of a prior probability distribution $p(\boldsymbol{\theta}_\mathrm{z}^k)$ over the model parameters $\boldsymbol{\theta}_\mathrm{z}^k$ is used. This distribution is chosen as normal-inverse-Wishart (NIW), given by

$$\begin{aligned} p(\boldsymbol{\theta}_\mathrm{z}^k) &= \mathrm{NIW}(\boldsymbol{\mu}_\mathrm{z}^k, \boldsymbol{\Sigma}_\mathrm{z}^k | k_0, \mathbf{m}_0^k, v_0, \mathbf{S}_0^k) \\ &= \mathcal{N}\left(\boldsymbol{\mu}_\mathrm{z}^k \middle| \mathbf{m}_0^k, \frac{1}{k_0}\boldsymbol{\Sigma}_\mathrm{z}^k\right) \mathcal{W}^{-1}(\boldsymbol{\Sigma}_\mathrm{z}^k | v_0, \mathbf{S}_0^k) \ , \end{aligned} \quad (6)$$

where the inverse-Wishart distribution $\mathcal{W}^{-1}(\cdot|\cdot)$ allows closed-form updates for the training process using an expectation maximization (EM) algorithm. The parameters $\mathbf{m}_0^k$, $\mathbf{S}_0^k$, k_0, and v_0 in (6) are discussed in the next section. The NIW distribution is commonly employed when the mean and covariance matrix of a normal distribution are assumed to be unknown [37].

The objective target function for training is chosen as the data likelihood distribution of all trials $p(\mathcal{S}^k|\boldsymbol{\theta}_\mathrm{z}^k)$ for a set of parameters $\boldsymbol{\theta}_\mathrm{z}^k$ which are learned from the series of demonstrations $\mathcal{S}^k$ for sub-model k. This distribution is given by marginalization over the latent space

$$p(\mathcal{S}^k|\boldsymbol{\theta}_\mathrm{z}^k) = \prod_{j=0}^{N^k} \int p(\mathbf{z}^{k,j}|\boldsymbol{\theta}_\mathrm{z}^k) \prod_{i=0}^{N^{k,j}} p(\mathbf{y}_i^{k,j}|\mathbf{z}^{k,j}, \mathbf{x}_i^{k,j}) \, \mathrm{d}\mathbf{z}^{k,j} \ , \quad (7)$$

in which the integral can be solved in closed form under the assumed distributions.

B. Model Training

The parameters $\boldsymbol{\theta}_\mathrm{z}^k$ are learned using the EM algorithm proposed in [19]. In the E-step of the algorithm, an estimation of the posterior distribution parameters $\boldsymbol{\theta}_\mathrm{z}^k = \{\boldsymbol{\mu}_\mathrm{z}^k, \boldsymbol{\Sigma}_\mathrm{z}^k\}$ for all trials $j = 1, \ldots, N^k$, of the series $k = 1, \ldots, N$, is computed. In the M-step, a new estimate $\hat{\boldsymbol{\theta}}_\mathrm{z}^k = \{\hat{\boldsymbol{\mu}}_\mathrm{z}^k, \hat{\boldsymbol{\Sigma}}_\mathrm{z}^k\}$ is found by maximizing the evidence lower bound (ELBO) of the data log-likelihood distribution $\log p(\mathcal{S}^k|\boldsymbol{\theta}_\mathrm{z}^k)$. This M-step can be executed using closed-form expressions [19].

In this work, the EM algorithm is initialized by the optimal parameters without prior knowledge, i.e., $k_0 = 0$, $\mathbf{m}_0 = \hat{\boldsymbol{\mu}}_\mathrm{z}^{k,*}$, $\hat{\boldsymbol{\Sigma}}_\mathrm{z}^k = \hat{\boldsymbol{\Sigma}}_\mathrm{z}^{k,*}$, and $\hat{\boldsymbol{\Sigma}}_\mathrm{y}^k = \hat{\boldsymbol{\Sigma}}_\mathrm{y}^{k,*}$, where the asterisk (*) indicates the maximum likelihood estimate (MLE) of the respective parameter. The further iterations are performed by alternating the E- and the M-step using the maximum a posteriori (MAP) estimate with $k_0 > 0$ and setting $v_0 = \dim(\mathbf{z}^k) + 1$ and $\mathbf{S}_0^k = (v_0 + N_\mathrm{z} + 1)\mathrm{blockdiag}(\hat{\boldsymbol{\Sigma}}_\mathrm{z}^{k,*})$.

Using the above training procedure, N sub-models are trained and the parameters $\boldsymbol{\theta}_\mathrm{z}^k$, $k = 1, \ldots, N$ are computed. Additionally, a cumulative model with index $k = 0$ is trained with the same procedure using the complete dataset $\mathcal{S}$, i.e., all available trials $\mathcal{T}^j$, $j = 1, \ldots, N^k$, from all series $\mathcal{S}^k$, $k = 1, \ldots, N$. Hence, a total of $N + 1$ sub-models are available for the generation of new tool motions.

C. Generation of New Tool Motions

Given a new path $\mathcal{P}$ on the surface of a different object with similar geometry, the ProSIP is utilized in this section to generate the corresponding tool motion. The path $\mathcal{P} = \{\boldsymbol{\eta}_i^\mathrm{p}, \mathbf{x}_i^\mathrm{p}, \mathbf{c}_i^\mathrm{p}\}_{i=1}^{N_\mathrm{p}}$ consists of N_p path points, where each point is described by the pose $\boldsymbol{\eta}_i^\mathrm{p}$ of the contact frame $\mathcal{C}$ w.r.t. the object frame $\mathcal{O}$, see Fig. 2, the surface feature vectors $\mathbf{x}_i^\mathrm{p}$, and the context vectors $\mathbf{c}_i^\mathrm{p}$. The result of the tool motion generation is the tool path $\mathcal{T}^\mathrm{p} = \{\boldsymbol{\xi}_i^\mathrm{p}\}_{i=1}^{N_\mathrm{p}}$, consisting of the poses $\boldsymbol{\xi}_i^\mathrm{p}$ of the tool frame $\mathcal{T}$ w.r.t. the contact frame $\mathcal{C}$ for all path points contained in $\mathcal{P}$.

First, the $N + 1$ sub-models (1) are conditioned on the path's feature vectors $\mathbf{x}_i^\mathrm{p}$ and the context vectors $\mathbf{c}_i^\mathrm{p}$ by setting unknown tool pose vectors $\boldsymbol{\xi}_i^\mathrm{p}$ contained in the observation vectors $\mathbf{y}_i^\mathrm{p}$ to zero. To this end, a single step of the EM algorithm [19] is executed to maximize the data log-likelihood distribution (7), and new parameter estimates $\boldsymbol{\theta}_\mathrm{z}^{k,\mathrm{p}} = \{\boldsymbol{\mu}_\mathrm{z}^{k,\mathrm{p}}, \boldsymbol{\Sigma}_\mathrm{z}^{k,\mathrm{p}}\}$ of the posterior distributions of the latent space variable $p(\mathbf{z}^{k,\mathrm{p}}|\boldsymbol{\theta}_\mathrm{z}^{k,\mathrm{p}}, \mathcal{P}, \hat{\boldsymbol{\Sigma}}_\mathrm{y}^k)$ are obtained.

Next, the decoder distribution

$$p(\mathbf{y}_i^{k,\mathrm{p}}|\mathbf{z}^{k,\mathrm{p}}, \mathbf{x}_i^\mathrm{p}) = \mathcal{N}(\mathbf{y}_i^{k,\mathrm{p}}|\boldsymbol{\Phi}^k(\mathbf{x}_i^{k,\mathrm{p}})\mathbf{z}^{k,\mathrm{p}}, \boldsymbol{\Sigma}_{\mathrm{y},i}^{k,\mathrm{p}}) \quad (8)$$

of all sub-models $k = 0, \ldots, N$ in (1) is marginalized, which yields [19]

$$\boldsymbol{\mu}_{\mathrm{y},i}^{k,\mathrm{p}} = \begin{bmatrix} \boldsymbol{\mu}_\xi^{k,\mathrm{p}} \\ \boldsymbol{\mu}_\mathrm{c}^{k,\mathrm{p}} \end{bmatrix} = \boldsymbol{\Phi}^k(\mathbf{x}_i^\mathrm{p})\boldsymbol{\mu}_\mathrm{z}^{k,\mathrm{p}} \quad (9)$$

$$\begin{aligned} \boldsymbol{\Sigma}_{\mathrm{y},i}^{k,\mathrm{p}} &= \boldsymbol{\epsilon}^\mathrm{p}(\boldsymbol{\epsilon}^\mathrm{p})^\mathrm{T} + \boldsymbol{\Phi}^k(\mathbf{x}_i^\mathrm{p})\boldsymbol{\Sigma}_\mathrm{z}^{k,\mathrm{p}} \left(\boldsymbol{\Phi}^k(\mathbf{x}_i^\mathrm{p})\right)^\mathrm{T} \\ &= \begin{bmatrix} \boldsymbol{\Sigma}_{\xi,\xi,i}^{k,\mathrm{p}} & \boldsymbol{\Sigma}_{\xi,\mathrm{c},i}^{k,\mathrm{p}} \\ \boldsymbol{\Sigma}_{\mathrm{c},\xi,i}^{k,\mathrm{p}} & \boldsymbol{\Sigma}_{\mathrm{c},\mathrm{c},i}^{k,\mathrm{p}} \end{bmatrix} \end{aligned} \quad (10)$$

$$\boldsymbol{\Sigma}_\mathrm{y}^{k,\mathrm{p}} = \frac{1}{N^\mathrm{p}} \sum_{i=1}^{N_\mathrm{p}} \boldsymbol{\Sigma}_{\mathrm{y},i}^{k,\mathrm{p}} \ , \quad (11)$$

where $\boldsymbol{\mu}_{\xi,i}^{k,\mathrm{p}}$ represents the mean tool pose at the path point i for the conditioned sub-model k, and its uncertainty is described by the covariance matrix $\boldsymbol{\Sigma}_{\xi,\xi,i}^{k,\mathrm{p}}$.

Remark 2: The integral marginalization can be calculated in closed form because of the assumed model structure, the distribution of the latent space variable $\mathbf{z}^k$, and their prior distribution of the parameters. However, in general this is not possible. This limitation can be handled by using a Monte-Carlo method to approximate the distribution or by learning the decoder distribution directly through a variational autoencoder (VAE).

The $\boldsymbol{\epsilon}^\mathrm{p}$ allows the incorporation of uncertainty along the conditioned path, e.g., by incorporating allowed process or measurement uncertainty into the generated tool motion. It allows for putting more trust into the observation than the learned models and vice versa.

Finally, based on the mean tool poses $\boldsymbol{\mu}_{\xi,i}^{k,\mathrm{p}}$ of all sub-models, the tool motion $\boldsymbol{\xi}_i^\mathrm{p}$ is computed as the weighted aggregation with the data likelihood distributions (7) $p(\boldsymbol{\mu}_{\mathrm{y},i}^{k,\mathrm{p}}|\mathbf{z}^k, \mathbf{x}_i^\mathrm{p}) = p(\boldsymbol{\mu}_{\mathrm{y},i}^{k,\mathrm{p}}|\hat{\boldsymbol{\mu}}_{\mathrm{y},i}^k, \hat{\boldsymbol{\Sigma}}_{\mathrm{y},i}^k)$ in the form

$$\boldsymbol{\xi}_i^\mathrm{p} = \frac{\sum_{k=0}^N p(\boldsymbol{\mu}_{\mathrm{y},i}^{k,\mathrm{p}}|\hat{\boldsymbol{\mu}}_{y,i}^k, \hat{\boldsymbol{\Sigma}}_{y,i}^k)\boldsymbol{\mu}_{\xi,i}^{k,\mathrm{p}}}{\sum_{k=0}^N p(\boldsymbol{\mu}_{\mathrm{y},i}^{k,\mathrm{p}}|\hat{\boldsymbol{\mu}}_{y,i}^k, \hat{\boldsymbol{\Sigma}}_{y,i}^k)} \ , \ i = 1, \ldots, N_\mathrm{p} \ , \quad (12)$$

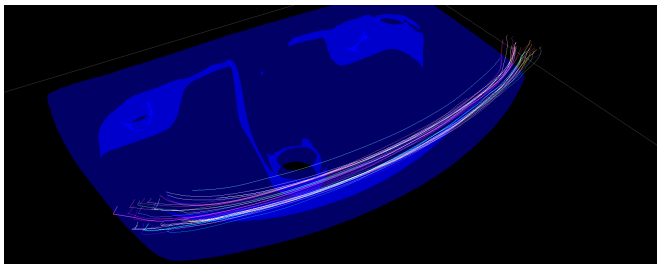

Fig. 5: Visualization of the tool paths in the training dataset on the front edge of a bathroom sink.

where $\hat{\boldsymbol{\theta}}_{\mathrm{z}}^{k}$ are the parameters of the learned $N+1$ sub-models. By marginalization of the decoder distributions (8) over $\mathbf{z}^k$ of each sub-model, the parameters $\boldsymbol{\theta}_{\mathrm{y},i}^{k} = \{\hat{\boldsymbol{\mu}}_{\mathrm{y},i}^{k}, \hat{\boldsymbol{\Sigma}}_{\mathrm{y},i}^{k}\}$ are calculated.

D. Trajectory Generation

The Cartesian motion of the tool frame $\mathcal{T}$ w.r.t. the object frame $\mathcal{O}$ is found by composing the two poses $\boldsymbol{\eta}_i^{\mathrm{p}}$ and $\boldsymbol{\xi}_i^{\mathrm{p}}$ as homogeneous transformations $\mathbf{H}_i^{\mathrm{p}} = \mathbf{H}(\boldsymbol{\eta}_i^{\mathrm{p}})\mathbf{H}(\boldsymbol{\xi}_i^{\mathrm{p}})$. Based on the newly conditioned tool motion and the homogenous transformation between the robotic platform base $\mathcal{R}$ and object $\mathcal{O}$, the task-space motion for the robotic platform is computed. For instance, a task-space controller can execute this task-space motion [38]. Further optimization techniques can be utilized to generate an optimal joint-space trajectory for a given robot model. Approaches such as time-optimal trajectory planning or time-optimal path parametrization [34], [39]–[41] combined with a computed torque controller allow the execution of the optimized trajectory.

VI. Results

In this section, the ProSIP framework is applied to an edge-cleaning task of bathroom sinks. First, the collected dataset (see Section IV) and trained models (see Section V) for the trajectory generation of the tool frame $\mathcal{T}$ are detailed. Then, the robotic task execution is demonstrated in simulation and for a real experiment using the 9-DoF robotic system shown in Fig. 1. A video of the simulations and the experimental demonstrations is found at www.acin.tuwien.ac.at/a79d.

A. Dataset and Models

The training dataset consists of four series of trials, $\mathcal{S}^1, \ldots, \mathcal{S}^4$, $N = 4$, demonstrating the cleaning of the front edge of a given bathroom sink with left-to-right and right-to-left motions and different tool orientations. In total, 45 trials are recorded, see Fig. 5, resulting in 202 306 data points. For each of the series, Gaussian sub-models (1) with the indices $k = 1, \ldots, N$ are trained. Additionally, the combined model with the index $k = 0$ is trained utilizing all trials in the dataset $\mathcal{S}$, see Section V-B, which is beneficial for generalization to new, unseen object geometries. In total, five models are used. The feature vector $\mathbf{x}_i^{k,j}$ of the kernel function $\psi_{d,h,g}^{k}(\mathbf{x})$ in (4) is set to the mean curvature $\frac{1}{2}(\kappa_{i,1}^{k,j} + \kappa_{i,2}^{k,j})$. The centers of the RBF $\mathbf{r}_{d,h,g}^{k}$ in (4) are evenly distributed across the training data $\mathcal{S}^k$, $k = 1, \ldots, 4$. Additionally, the variances $\sigma_{d,h,g}^{k}$ are determined based on the variance of the mean curvature in the training data.

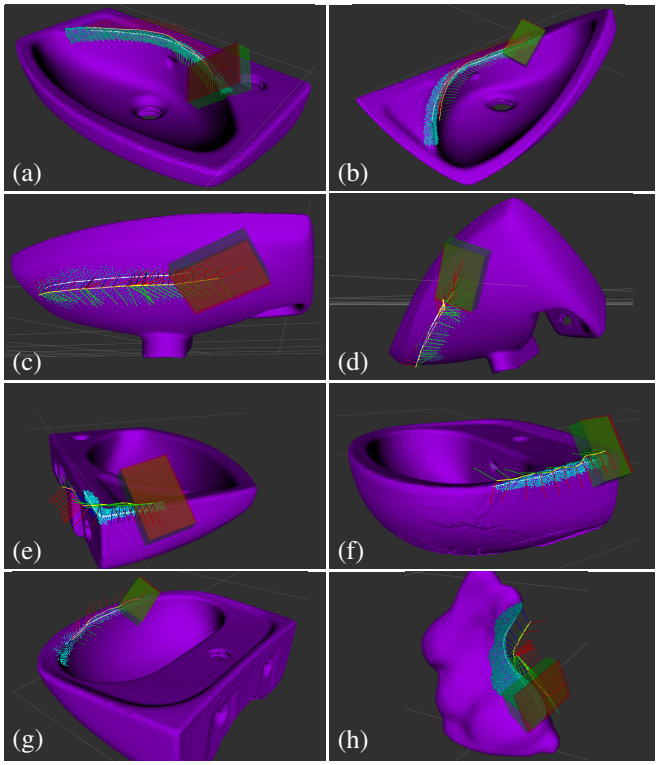

Fig. 6: Eight examples for the edge-cleaning task using different surface paths on different object geometries. The white line is the input surface path, while the yellow path with the sequence of frames illustrates the output tool motion. The sponge is visualized as green transparent box. The cyan dots indicate areas that are cleaned during the tool motion.

B. Simulation Results

The ProSIP framework is demonstrated in simulation on different edges of different bathroom sinks with similar geometry. To this end, new surface paths on different edges of the 3D objects are generated by manually picking vertices of the mesh and connecting them using exact geodesic paths [42]. These paths are used to generate the tool motions according to Section V-C and are executed in simulation using a compliant task-space controller, see Section V-D. The controller stiffness is specified w.r.t. the tool frame $\mathcal{T}$ [35], allowing compliant behavior along the surface normal vector. The interaction between the robot and the 3D object is simulated using the physics simulator Mujoco [43].

1) Bathroom Sinks with Similar Geometry: Figure 6 illustrates the result of the ProSIP framework for different paths on different object geometries. Each image (a)–(h) shows the input surface path as a white line and the output tool path as a yellow line. Additionally, the tool frame $\mathcal{T}$ is visualized for multiple points along the path, and the sponge is shown as a green transparent box for the first point. Cyan dots mark the areas cleaned by the tool during the cleaning motion.

The different examples in Fig. 6 highlight the ability of ProSIP to generalize to edges of different curvature on the outside and the inside of the sink and to paths with a corner. Different bathroom sinks and a rabbit-shaped object are used to demonstrate the conditioning on various object geometries

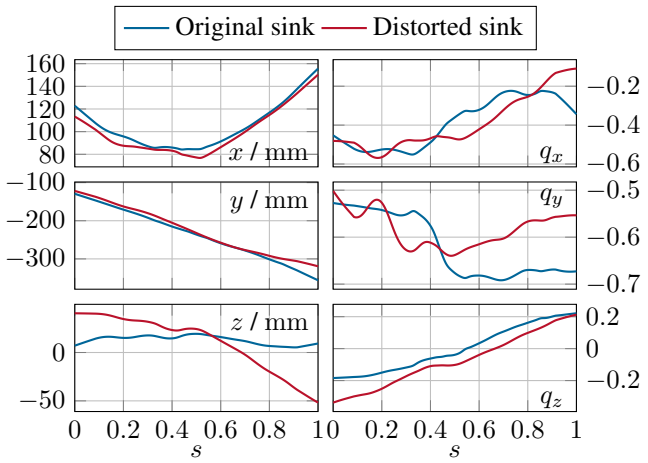

Fig. 7: Tool paths corresponding to the edge-cleaning task shown in Fig. 6a (original sink) and Fig. 6b (distorted sink).

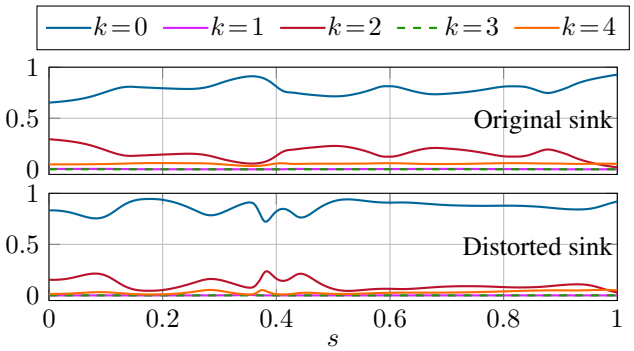

Fig. 8: Likelihoods $p(\boldsymbol{\mu}_{\mathrm{y},i}^{k,\mathrm{p}}|\mathbf{z}^k, \mathbf{x}_i^{\mathrm{p}})$ of the individual models with the indices $k = 0, \ldots, 4$, which are used in the weighted aggregation (12).

outside the training dataset distribution. Furthermore, the generated tool motions $\boldsymbol{\xi}_i^{\mathrm{p}}$ from (12) are smooth regarding position and orientation; see Fig. 7 for the tool paths of Fig. 6a and Fig. 6b.

2) Distorted 3D Models of Bathroom Sinks: In this section, significantly distorted 3D models of bathroom sinks with higher geometric complexity are investigated. Each distorted 3D model is obtained by torsion around the z- and x-axis by $45°$, see Fig. 6b and Fig. 6d. The surface paths from the original sink are directly transferred to the distorted sink via the vertex and face indices of the object meshes. As an example, the tool paths of Fig. 6a and Fig. 6b are shown as a function of the path position $s \in \{0, 1\}$ in Fig. 7. The output path for the transformed object is smooth and qualitatively similar to the path of the original object. Additionally, the underlying probabilities $p(\boldsymbol{\mu}_{\mathrm{y},i}^{k,\mathrm{p}}|\mathbf{z}^k, \mathbf{x}_i^{\mathrm{p}})$ of (12) for all model indices $k = 0, \ldots, 4$ are depicted in Fig. 8 as a function of s. The most significant contribution to the output tool path originates from the combined model with the index $k = 0$, which emphasizes the importance of this model. The second major contribution stems from the model $k = 2$, a series of right-to-left edge-cleaning motions. This corresponds to the input surface path in Fig. 6a and Fig. 6b.

3) Validation in Simulation: In the simulation, the generated tool motions are validated by evaluating the intersection between the object mesh and the mesh of the sponge tool, represented by the red and green boxes in Fig. 6. The centers of all faces, which are entirely covered by the mesh of the sponge tool, are marked with a cyan dot. This visualization determines the cleaned surface by the sponge. It reveals that the tool stays in good contact with the object's surface throughout the motion, i.e., significant intersections with the green box, while no collisions occur with the mounting plate of the sponge, i.e., the red box. Additionally, the cleaned surface area is determined by combining all faces intersecting with the sponge; see the supplementary video. This result also confirms that ProSIP implicitly has learned the correct tool orientation and distance to the surface.

C. Experimental Results

The experimental validation is performed on the 9-DoF robotic system shown in Fig. 1. Dirt spots on the surface of the bathroom sink are applied using a whiteboard marker, which must be cleaned by the robotic system using the tool motions generated by the ProSIP framework and the compliant task-space controller. The coordinate frame of object $\mathcal{O}$ is calibrated in the robot's frame using the calibration method [44]. The experimental results are summarized in the supplementary video. It is demonstrated that the robot can execute the generated tool motions and successfully perform the cleaning task.

VII. Conclusions

This work proposes the framework probabilistic surface interaction primitives (ProSIP), which extend probabilistic motion primitives (ProMP) to continuous interaction tasks between a tool and a freeform 3D surface. ProSIPs are by design independent of time and invariant w.r.t. rigid-body displacements, as they systematically consider the projected path on the surface and the local surface features during the learning procedure. Human demonstrations are recorded using an instrumented tool, i.e., a standard tool with integrated sensors that allows seamless transfer to any robotic platform.

As an example task, the framework is employed to clean the edges of bathroom sinks. Demonstration trials are recorded for cleaning the front edge of the sink using a simple sponge tool only. Results show that the learned model can be generalized to different edges on different object geometries, including corners and significantly distorted objects. The model implicitly learned the correct tool orientation and position w.r.t. the freeform 3D surface. The generated tool motions are validated in simulation by evaluating the mesh intersections between the surface and the tool and in the experimental setup by cleaning dirt spots from the surface of the bathroom sinks.

Practically, ProSIPs capture the complex interaction between a tool and a freeform 3D surface, described by the contact point moving along a surface path. Therefore, the framework applies to many different applications, e.g., drawing, polishing, and sanding, with a suitable instrumented tool for recording the demonstrations.

Future works focus on building hierarchical models to systematically integrate different object geometries, e.g., planar

areas, edges, and corners, and different signals, e.g., forces and torques. Additionally, developing fully automatic calibration routines utilizing the CAD data of the instrumented tools and the demonstration environment will allow for easy adoption of the ProSIP framework in practical applications.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the financial support of Festo AG & Co. KG.

REFERENCES

[1] Z. Liu, Q. Liu, W. Xu, L. Wang, and Z. Zhou, "Robot learning towards smart robotic manufacturing: A review," *Robotics and Computer-Integrated Manufacturing*, vol. 77, p. 102360, 2022.

[2] F. Suárez-Ruiz, X. Zhou, and Q.-C. Pham, "Can robots assemble an IKEA chair?" *Science Robotics*, vol. 3, no. 17, p. eaat6385, 2018.

[3] G. Maeda, G. Neumann, M. Ewerton, R. Lioutikov, O. Kroemer, and J. Peters, "Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks," *Autonomous Robots*, vol. 41, pp. 593–612, 2017.

[4] G. A. Zachiotis, G. Andrikopoulos, R. Gornez, K. Nakamura, and G. Nikolakopoulos, "A survey on the application trends of home service robotics," in *Proc. ROBIO*, 2018, pp. 1999–2006.

[5] S. Frennert, H. Aminoff, and B. Östlund, "Technological frames and care robots in eldercare," *International Journal of Social Robotics*, vol. 13, pp. 311–325, 2021.

[6] Z. Zhu and H. Hu, "Robot learning from demonstration in robotic assembly: A survey," *Robotics*, vol. 7, no. 2, pp. 1–25, 2018.

[7] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: Challenges, representations, and algorithms," *Journal of Machine Learning Research*, vol. 22, no. 30, pp. 1–82, 2021.

[8] S. Kim, A. Shukla, and A. Billard, "Catching objects in flight," *IEEE Transactions on Robotics*, vol. 30, no. 5, pp. 1049–1065, 2014.

[9] F. Tian, C. Lv, Z. Li, and G. Liu, "Modeling and control of robotic automatic polishing for curved surfaces," *CIRP Journal of Manufacturing Science and Technology*, vol. 14, pp. 55–64, 2016.

[10] J. Li, T. Zhang, X. Liu, Y. Guan, and D. Wang, "A survey of robotic polishing," in *Proc. IEEE ROBIO*, 2018, pp. 2125–2132.

[11] 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, 2020.

[12] D. Martínez, G. Alenyà, and C. Torras, "Planning robot manipulation to clean planar surfaces," *Engineering Applications of Artificial Intelligence*, vol. 39, pp. 23–32, 2015.

[13] N. Vuković, M. Mitić, and Z. Miljković, "Trajectory learning and reproduction for differential drive mobile robots based on GMM/HMM and dynamic time warping using learning from demonstration framework," *Engineering Applications of Artificial Intelligence*, vol. 45, pp. 388–404, 2015.

[14] A. X. Lee, H. Lu, A. Gupta, S. Levine, and P. Abbeel, "Learning force-based manipulation of deformable objects from multiple demonstrations," in *2015 IEEE ICRA*, 2015, pp. 177–184.

[15] M. Ikeda, M. Ganglbauer, P. Ashok, S. Maddukuri, M. Hofmann, and A. Pichler, "Instrumented tool based robot programming - Parameterization of screwing process macros," *Procedia Manufacturing*, vol. 38, pp. 415–422, 2019.

[16] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[17] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 297–330, 2020.

[18] A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann, "Probabilistic movement primitives," in *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[19] S. Gomez-Gonzalez, G. Neumann, B. Schölkopf, and J. Peters, "Adaptation and robust learning of probabilistic movement primitives," *IEEE Transactions on Robotics*, vol. 36, no. 2, pp. 366–379, 2020.

[20] S. Stark, J. Peters, and E. Rueckert, "Experience reuse with probabilistic movement primitives," in *Proc. IEEE/RSJ IROS*, 2019, pp. 1210–1217.

[21] F. Frank, A. Paraschos, P. van der Smagt, and B. Cseke, "Constrained probabilistic movement primitives for robot trajectory adaptation," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2276–2294, 2022.

[22] L. Rozo and V. Dave, "Orientation probabilistic movement primitives on riemannian manifolds," in *Proc. CoRL*, 2022, pp. 373–383.

[23] M. Müller, "Dynamic time warping," in *Information Retrieval for Music and Motion*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 69–84.

[24] A. Gams, T. Petrič, M. Do, B. Nemec, J. Morimoto, T. Asfour, and A. Ude, "Adaptation and coaching of periodic motion primitives through physical and visual interaction," *Robotics and Autonomous Systems*, vol. 75, pp. 340–351, 2016.

[25] T. Kulak, J. Silvério, and S. Calinon, "Fourier movement primitives: an approach for learning rhythmic robot skills from demonstrations." in *Proc. RSS*, 2020.

[26] L. Rozo, M. Guo, A. G. Kupcsik, M. Todescato, P. Schillinger, M. Giftthaler, M. Ochs, M. Spies, N. Waniek, P. Kesper, and M. Burger, "Learning and sequencing of object-centric manipulation skills for industrial tasks," in *Proc. IEEE/RSJ IROS*, 2020, pp. 9072–9079.

[27] Y. Huang, L. Rozo, J. Silvério, and D. G. Caldwell, "Kernelized movement primitives," *The International Journal of Robotics Research*, vol. 38, no. 7, pp. 833–852, 2019.

[28] T. Ren, A. I. Cowen-Rivers, H. B. Ammar, and J. Peters, "Learning geometric constraints in task and motion planning," *arXiv preprint arXiv:2201.09612*, 2022.

[29] Y. Wang, C. Chen, F. Peng, Z. Zheng, Z. Gao, R. Yan, and X. Tang, "AL-ProMP: Force-relevant skills learning and generalization method for robotic polishing," *Robotics and Computer-Integrated Manufacturing*, vol. 82, p. 102538, 2023.

[30] H. Ben Amor, G. Neumann, S. Kamthe, O. Kroemer, and J. Peters, "Interaction primitives for human-robot cooperation tasks," in *Proc. IEEE ICRA*, 2014, pp. 2831–2837.

[31] A. Paraschos, E. Rueckert, J. Peters, and G. Neumann, "Model-free probabilistic movement primitives for physical interaction," in *Proc. IEEE/RSJ IROS*, 2015, pp. 2860–2866.

[32] X. Xue, L. Zuo, and N. Wang, "A robot human-like learning framework applied to unknown environment interaction," *Complexity*, vol. 2022, p. 5648826, 2022.

[33] Y.-Y. Lin, C.-C. Ni, N. Lei, X. Gu, and J. Gao, "Robot coverage path planning for general surfaces using quadratic differentials," *Proc. IEEE ICRA*, pp. 5005–5011, 2017.

[34] T. Weingartshofer, B. Bischof, M. Meiringer, C. Hartl-Nesic, and A. Kugi, "Optimization-based path planning framework for industrial manufacturing processes with complex continuous paths," *Robotics and Computer-Integrated Manufacturing*, vol. 82, p. 102516, 2023.

[35] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. London: Springer, 2010.

[36] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning.*, ser. Adaptive computation and machine learning. Cambridge: MIT Press, 2006.

[37] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge: MIT Press, 2012.

[38] H. Sadeghian, L. Villani, M. Keshmiri, and B. Siciliano, "Task-space control of robot manipulators with null-space compliance," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 493–506, 2014.

[39] J. Kim and E. A. Croft, "Online near time-optimal trajectory planning for industrial robots," *Robotics and Computer-Integrated Manufacturing*, vol. 58, pp. 158–171, 2019.

[40] L. Lu, J. Zhang, J. Y. H. Fuh, J. Han, and H. Wang, "Time-optimal tool motion planning with tool-tip kinematic constraints for robotic machining of sculptured surfaces," *Robotics and Computer-Integrated Manufacturing*, vol. 65, p. 101969, 2020.

[41] H. Pham and Q.-C. Pham, "A new approach to time-optimal path parameterization based on reachability analysis," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 645–659, 2018.

[42] J. S. B. Mitchell, D. M. Mount, and C. H. Papadimitriou, "The discrete geodesic problem," *SIAM Journal on Computing*, vol. 16, no. 4, pp. 647–668, 1987.

[43] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proc. IEEE/RSJ IROS*, 2012, pp. 5026–5033.

[44] T. Weingartshofer, C. Hartl-Nesic, and A. Kugi, "Automatic and flexible robotic drawing on complex surfaces with an industrial robot," *IEEE Transactions on Control Systems Technology*, pp. 1–14, 2023.