PRIMP: PRobabilistically-Informed Motion Primitives for Efficient Affordance Learning From Demonstration

Sipu Ruan, *Member, IEEE*, Weixiao Liu, *Member, IEEE*, Xiaoli Wang, *Graduate Student Member, IEEE*, Xin Meng, *Graduate Student Member, IEEE*, and Gregory S. Chirikjian, *Fellow, IEEE*

***Abstract*—This article proposes a Learning-from-Demonstration (LfD) method using probability densities on the workspaces of robot manipulators. The method, named PRobabilistically-Informed Motion Primitives (PRIMP), learns the probability distribution of the end effector trajectories in the 6-D workspace that includes both positions and orientations. It is able to adapt to new situations such as novel via points with uncertainty and a change of viewing frame. The method itself is robot-agnostic, in that the learned distribution can be transferred to another robot with the adaptation to its workspace density. Workspace-STOMP, a new version of the existing STOMP motion planner, is also introduced, which can be used as a postprocess to improve the performance of PRIMP and any other reachability-based LfD method. The combination of PRIMP and Workspace-STOMP can further help the robot avoid novel obstacles that are not present during the demonstration process. The proposed methods are evaluated with several sets of benchmark experiments. PRIMP runs more than five times faster than existing state-of-the-art methods while generalizing trajectories more than twice as close to both the demonstrations and novel desired poses. They are then combined with our lab's robot imagination method that learns object affordances, illustrating the applicability to learn tool use through physical experiments.**

***Index Terms*—Learning from demonstration (LfD), motion and path planning, probability and statistical methods, service robots.**

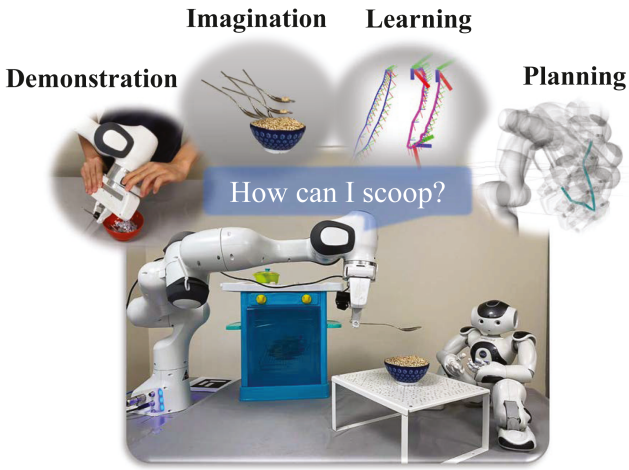


Fig. 1. Illustration for the general idea of this work. The robot is asked to use a spoon to scoop from a bowl in a household environment. With the help of human demonstrations, imagination of object affordance, learning skills from the demonstrations, and motion planning, the robot is able to fulfill the task in a novel scene with previously unseen obstacles.

I. Introduction

FOR a robot to be truly intelligent, it needs the ability to learn from prior knowledge while adapting to previously unseen scenarios. The prior knowledge can be previously feasible trajectories that are hard-coded, like in a structured factory environment. Prior knowledge can also be from human-demonstrated motions, which are difficult to preprogram but are ubiquitous in household environments, such as scooping powder (as in Fig. 1). The latter case is much more challenging and related to a popular field in robot learning, namely *Learning-from-Demonstration* (LfD) [1] or *Programming-by-Demonstration* [2]. Many works on LfD encode demonstrations as trajectories in Euclidean space [3], [4]. For example, the angles and velocities of all joints are considered as a multidimensional state vector; or only the positions of the end effector are modeled. However, the models in the full workspace (including both position and orientation) of a robot manipulator have not been considered until recently [5], [6], [7], [8], [9]. Our work focuses on the robot workspace and proposes a novel method using probability densities on Lie groups, denoted as *PRobabilistically-Informed Motion Primitives* (PRIMP). The mathematical model is inspired by a concept initially introduced in our group more than 25 years ago and more recent works on loop entropy [10],

Manuscript received 22 November 2023; accepted 1 April 2024. Date of publication 16 April 2024; date of current version 17 May 2024. This paper was recommended for publication by Associate Editor Matthew Walter and Editor Jeannette Bohg upon evaluation of the reviewers' comments. This work was supported in part by the NUS Startup Grant A-0009059-02-00 and Grant A-0009059-03-00, in part by the CDE Board Account E-465-00-0009-01, in part by the National Research Foundation, Singapore, under its Medium Sized Centre Programme—Centre for Advanced Robotics Technology Innovation (CARTIN), under Subaward A-0009428-08-00, and in part by the AME Programmatic Fund Project MARIO under Grant A-0008449-01-00. *(Corresponding author: Gregory S. Chirikjian.)*

Sipu Ruan, Xiaoli Wang, and Xin Meng are with the Department of Mechanical Engineering, National University of Singapore, Singapore 117575.

Weixiao Liu is with the Department of Mechanical Engineering, Johns Hopkins University, Baltimore, MD 21218 USA.

Gregory S. Chirikjian is with the Department of Mechanical Engineering, National University of Singapore, Singapore 117575, and also with the Department of Mechanical Engineering, University of Delaware, Newark, DE 19716 USA (e-mail: gchirik@udel.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TRO.2024.3390052, provided by the authors.

Digital Object Identifier 10.1109/TRO.2024.3390052

[11] and inverse reachability mapping [12], [13]. It is further extended here to LfD with via-point conditioning as compared to only subjecting to end constraints. The learned knowledge is robot-agnostic but can be adapted to the workspace of a specific robot. It is also able to deal with extrapolation cases and model with a few or even a single demonstration.

Demonstrations are typically conducted in a scene with only the robot and the object with which it interacts. However, when there are new obstacles that are not present during the demonstrations, only using LfD methods is not enough to generate feasible trajectories. Here is where the motion planning process fits in. It is able to guarantee safe and optimal motions for a task, which is formulated as motion constraints. For most motion planners, either sampling-based or optimization-based, constraints are added as a function with specific expressions. For example, to hold a cup that is filled with water upright, one usually needs to define the vector that is perpendicular to the cup opening to be parallel with the global z-axis. Such an expression is, however, not trivial to define for more complex tasks, e.g., scooping powders and writing letters. For such tasks, LfD can play the role of guiding the motion planners via a trajectory-wise cost function.

Guided motion planning, which combines LfD and motion planning, has become popular in the recent decade [14], [15], [16], [17]. The goal is to make the motion collision-free while keeping as many critical features from the learned trajectory as possible. In this work, an optimization-based planner, *Stochastic Trajectory Optimization for Motion Planning* (STOMP) [18], is applied as the base framework. The learned trajectory distribution via PRIMP is used as a reference. It is first treated as the initial condition for the planner. Then, a novel cost function with respect to this reference distribution is proposed to guide the planning process. Instead of joint space, the cost function is based on the workspace of the robot end effector. Therefore, the planner is named as *Workspace-STOMP*. This planner can be used as a postprocess of both PRIMP and any other reachability-based LfD method.

Apart from novel obstacles, the object that the robot interacts with might also be previously unseen. For example, the robot has been taught how to pour powders from a particular cup into a particular bowl. In this case, the cup is treated as a tool for pouring and the bowl has the affordance of containing. In a new scenario, the tool for pouring might be changed into a spoon and the object might become a vase, which has the same affordance of containing. The learned trajectory distribution should be able to adapt to this new situation with the same set of demonstrations, even when the appearances and categories of the tool and object are totally different. In order to fulfill similar tasks intelligently, the understanding of object functionality and affordance is a key aspect [19]. Recent works proposed methods to learn object affordance, such as pouring, via physics-based simulation [20], [21]. Our work applies this idea to learn the key points of a task, i.e., the pouring pose, which is used as a new goal or via point for reproduction. A new task that learns the affordance of scooping is proposed and implemented in the simulation environment. The affordance learning of an object using simulation is then combined into a robotic system with PRIMP and Workspace-STOMP in physical experiments.

The contributions of our work are listed as follows.[1]

1) An LfD method, PRIMP, is proposed to generate a reference workspace trajectory distribution for basic motion primitives.
2) By proposing a novel cost function related to the reference distribution, Workspace-STOMP is proposed to keep the shape of the trajectory similar while maintaining the feasibility of the plan.
3) A novel robotic system that combines LfD, motion planning, and affordance-based simulation is proposed and physically demonstrated in a robot manipulator platform.

The rest of this article is organized as follows. Section II reviews related literature on LfD and guided motion planning. Section III introduces the proposed PRIMP method. Section IV proposes the Workspace-STOMP motion planning algorithm. Then, Section V evaluates PRIMP and Workspace-STOMP through benchmarks with existing probabilistic methods for solving LfD and planning problems. Section VI proposes a robotic system that combines PRIMP and Workspace-STOMP with affordance-based simulation and demonstrates via several physical experiments. The proposed method is discussed in Section VII. Finally, Section VIII concludes this article.

II. Literature Review

LfD is a fast-growing field in robot learning. It aims for robots to imitate human experts while adapting to new situations. A growing literature has shown that the combination of LfD and motion planning is able to adapt to a broader field of scenarios where previously unseen obstacles appear. The following reviews related work on LfD methods and guided motion planning algorithms.

A. Learning From Demonstration

Over the past decade, many LfD methods have been proposed to encode trajectory models and generalize them to new situations, most of which stay in Euclidean space. Dynamical movement primitives (DMP) [22] learns a damped spring model to obtain a mapping between acceleration and position–speed pair. It is able to represent and encode goal-directed or periodic trajectories given any start and goal points but cannot generalize to pass via points. By combining DMP with Gaussian processes, uncertainty information can be exploited during reproduction [23]. The parameters, i.e., the global time scale, weights of the forcing term, etc., can be learned using reinforcement learning (RL) techniques [24]. To further solve the violation of joint limits caused by exploration noise, constrained DMP is proposed [25]. Recently, the DMP-based methods have been comprehensively reviewed in [26]. Methods based on the RL framework are also popular in learning primitive skills. A natural actor–critic framework is used to learn robot motor primitives in [27], demonstrating an order of magnitude improvements

[1]The code is available at https://github.com/ChirikjianLab/primp-matlab.

in both accuracy and speed of skill learning to the compared counterparts. It is applied in learning a high-speed motion such as hitting a ball. Further extensions to episodic policy search have been proposed to learn more complex skills such as swing-up motions [28].

Another class of LfD methods is based on probabilistic models, which is the category that our proposed method falls in.The task-parameterized Gaussian mixture model (TP-GMM) [29], [30] encodes the demonstrations using virtual spring–damper systems in multiple reference frames. It learns the parameters in an expectation–maximization fashion and is able to reproduce in new situations, such as adaptation to via points. DMP can also be reformulated in a probabilistic setting. For example, by combining with Gaussian mixture regressions (GMR), the parameters of DMP can be estimated with projection into different coordinate system to learn from different sets of demonstrations for a humanoid robot [31]. Using probabilistic linear dynamical system models, DMP can be formulated probabilistically, enabling Kalman filtering and smoothing to infer from sensor measurement during executions [32]. Probabilistic movement primitives (ProMP) [33] learns the distribution given a set of demonstrated trajectories and uses conditioning to generalize to novel situations such as via-points and temporal variances. The learning phase is based on a hierarchical Bayesian model, which requires the definition of basis such as radial and Fourier basis functions. Several variants have been proposed to estimate parameters using fewer training instances [34], unify adaptation of the trajectory using constrained optimization [35], adapt to external contextual information such as object mass using active learning technique [36], etc. However, when the data dimension is large, ProMP might require more basis functions to encode the demonstrations well. To deal with the extrapolation issue, via-point movement primitives (VMP) [37] combines ProMP with ideas from DMP by splitting the trajectory model into elementary trajectories and shape modulation terms. Kernelized movement primitives (KMP) [38] uses the kernel trick to deal with high dimensional input data without defining basis functions and can also cope with extrapolation issues. In our work, ProMP is selected in the benchmark studies to only handle the data in Euclidean space, i.e., the position information of trajectories.

Recently, more investigations have been conducted on encoding trajectories and generalizing them to new situations within a manifold, i.e., to precisely describe the orientation of the end effector [6], [39]. DMP is extended to encode orientation information, which uses quaternions to provide singularity-free representation [40]. A more general formulation working in Riemannian manifolds is proposed to encode and learn composite manifolds such as unit quaternions and symmetric positive-definite matrices [41]. Several probabilistic methods on Riemannian manifolds are also proposed. An extension for the Gaussian mixture model (GMM) with inference using GMR or TP-GMM has been proposed to work on Riemannian manifolds [5]. The operations such as Gaussian product and conditioning are defined using Riemannian statistics and exponential mapping. It also has applications for bimanual manipulation [42] and grasping [8] tasks. Operations in ProMP can also work on Riemannian manifolds, including trajectory modulation, blending, task parameterization, etc. [43]. It uses geodesic regression to estimate the weight distribution, which acts as a generalized technique for linear regression in Euclidean space. Probabilistic learning of quaternions is applied in the KMP framework for modeling orientation and angular velocity [7]. The trajectory after adaptation is then optimized with angular and acceleration constraints. Orientation-KMP is used in the benchmark studies. Our proposed LfD method differs in that no basis function or kernel is required. Also, the distribution is defined by the relative poses between adjacent time steps compared to the absolute poses in the trajectory.

B. Guided Motion Planning

Guided motion planning can be viewed as a special type of constrained motion planning. The constraints are mostly treated as measures of how close a trajectory is to the guiding reference. A demonstration-guided motion planning framework has been proposed to combine LfD and a sampling-based planner [15]. It encodes the reference in joint space with a probabilistic model and records surrounding landmarks for correspondence matching among different scenarios. The reference path is extended around new obstacles by sampling configurations from the probabilistic model and formulating a local graph to search for a new safe and optimal path. Combined LfD and motion planning carry out probabilistic inference to search optimal trajectories for a certain skill that is feasible in novel scenes [44]. The trajectory prior is generated by the Gaussian process, which is the solution to a linear time-varying stochastic differential equation. To reproduce trajectories, the algorithm uses maximum a posteriori (MAP) inference. The method is extended to conduct demonstrations in a cluttered environment with weighted skill learning [45]. Defining cost functions using learned trajectory from the LfD method is popular in guiding optimization-based motion planners. For example, the cost functional of covariant Hamiltonian optimization for motion planning (CHOMP) [46] is proposed to measure deviations from the probability distribution of the demonstrations [47]. Alternatively, results from ProMP have been used as initial conditions for CHOMP [48]. For a partially observable environment, the inverse RL technique is utilized to derive motion level guidance, the reward of which is used to guide the TrajOpt algorithm [49]. STOMP is another popular planner that uses rollouts sampled from probabilistic distribution to avoid gradient computations. Cost functions based on DMP working on joint space (namely GSTOMP) [16] and Cartesian constraints that deal with both position and orientation of end effector (i.e., Cartesian-STOMP) [17] have been proposed. This work uses Cartesian-STOMP for benchmarks, but the planning part of our method differs significantly. Instead of using only one trajectory as guidance, ours utilizes the information on the probabilistic distribution of the trajectories.

III. Probabilistically-Informed Motion Primitives

This section introduces the proposed LfD method, namely PRIMP. This is a probabilistic method to encode the trajectory that involves both position and orientation of the robot end effector. The trajectory is represented discretely by a number

of time steps (i.e., N_{step}). Each pose is modeled as an element in Lie group G. Therefore, the full state is considered in a product space $G \times \ldots \times G$, resulting in a state vector of dimension $6N_{\text{step}}$. Consider a set of demonstrated trajectories $\{g_0^{(k)}, g_1^{(k)}, \ldots, .g_n^{(k)}\}$, where $g_i^{(k)} \in \text{SE}(3)$ is the ith step in the kth demonstration[2]. The goal is to compute a probability distribution of the given demonstrations as a reference to guide the future executions of the robot for a similar task.

In this work, the widely-recognized group in robotics, i.e., SE(3), is used to derive the proposed method. The extension to another Lie group, i.e., *pose change group* (PCG(3)) [50], is also introduced. Without loss of generality and special mentions, SE(3) is used in the following introductions of the proposed method.

A. General Framework of PRIMP

First, all the demonstrated trajectories are temporally aligned into the same time scale, via *globally-optimal reparameterization algorithm* (GORA) [51] (see Section III-B). After alignment, the probability distribution of the $(i+1)$th pose with respect to the ith pose is approximated using a Lie-theoretic method (see Section III-C). The computed initial mean and covariance are then encoded into the joint distribution of the whole trajectory (see Section III-D). Several types of adaptations to novel scenarios are then introduced (see Section III-E), which are as follows.

1) When an intermediate pose is different from the given mean and has uncertainties, a posterior distribution is computed to adapt to this new situation (see Section III-E1).
2) When there is a change in the viewing frame, the encoded distribution can be adapted in the sense of equivariance (see Section III-E2).
3) When another robot is operated for the same task, the learned distribution can be further conditioned to the high-density region of the new workspace (see Section III-E3).

All the aforementioned computations can be extended to PCG(3), which is discussed in Section III-F.

B. Temporal Alignment for Demonstrations Using GORA

Different demonstrations might have different speeds of execution, even when the trajectories have the same shape [3]. Speed differences among demonstrations may cause the same feature to be parameterized at different time steps. This misalignment in the temporal axis makes probabilistic modeling difficult. Therefore, a temporal reparameterization to align all the demonstrated trajectories into the same time scale is necessary and essential. Existing works often used the Dynamic Time Warping (DTW) method to temporally align different demonstrations [4], [52], which choose one base trajectory and warp others into it. In principle, different choices of the base trajectory might give different alignment results. Alternatively, in our work, the variational calculus technique is applied to align multiple trajectories with global optimality in the temporal axis, which is named as GORA [51].

1) Problem Formulation: Suppose an SE(3) sequence $g(\tau)$ is parameterized by $\tau \in [0,1]$. Here, $\tau(t) : [0,1] \to [0,1]$ is a 1-D monotonic function that parameterizes time. The total variation of the whole sequence can be computed as the sum of the squared derivative with respect to time t, i.e.,

$$J = \int_0^1 f(\tau, \dot{\tau})\, dt \doteq \int_0^1 \mathfrak{g}(\tau)\, \dot{\tau}^2\, dt \tag{1}$$

where $\dot{\tau} = d\tau/dt$. The first half of (1) is a general form of the functional that defines the variational problem. The second half is a specially designed structure, so as to achieve a global optimality of the solution. The expression of $\mathfrak{g}(\tau)$ in the integrand is defined based on the body velocity of an SE(3) trajectory, i.e.,

$$\mathfrak{g}(\tau) \doteq \left\| g^{-1} \frac{\partial g}{\partial \tau} \right\|_W^2 \tag{2}$$

where $\|\cdot\|_W$ denotes the weighted Frobenius norm defined such that for any $\mathcal{A} \in \mathbb{R}^{4\times 4}$, $\|\mathcal{A}\|_W = \sqrt{\text{tr}(\mathcal{A}^T W \mathcal{A})}$, for a symmetric matrix

$$W \doteq \begin{pmatrix} \frac{1}{2}\text{tr}(I)\mathbb{I}_3 - I & \mathbf{0} \\ \mathbf{0}^T & 1 \end{pmatrix} \in \mathbb{R}^{4\times 4}$$

where I is the 3×3 diagonal inertia tensor corresponding to a solid sphere of unit mass and $\mathbb{I}_3$ is the 3×3 identity matrix. The structure of the function $f(\tau, \dot{\tau})$ is specifically chosen using (2) as the weighted norm of body velocity with respect to a temporal parameterized variable τ. After solving (1) with this design, the global optimality of the solution can be obtained.

2) Globally Optimal Solution and GORA: To extremize (1), the Euler–Lagrange equation is used, which has the form

$$\frac{\partial f}{\partial \tau} - \frac{d}{dt}\left(\frac{\partial f}{\partial \dot{\tau}}\right) = 0\,. \tag{3}$$

With a special structure of (2), the solution of (3) can be proved to be globally minimal, resulting in the following theorem.

Theorem III.1: The Euler–Lagrange (3) globally minimizes (1) if the following statements hold.

1) The integrand is of the form as (2).
2) $\tau : [0,1] \to [0,1]$ is monotonically increasing with boundary conditions $\tau(0) = 0$ and $\tau(1) = 1$.

The minimizer is $\tau^*(t) = F^{-1}(t)$, where $F^{-1}(\cdot)$ is the inverse function of

$$F(\tau^*) = \frac{\int_0^{\tau^*} \mathfrak{g}^{\frac{1}{2}}(\sigma)\, d\sigma}{\int_0^1 \mathfrak{g}^{\frac{1}{2}}(\sigma)\, d\sigma} = t\,. \tag{4}$$

The proof of Theorem III.1 can be referred to the work in [51] and [53]. Algorithm 1 summarizes the workflow of GORA for SE(3) sequences. Line 1 initializes the temporal parameter τ and the original time scale t uniformly within [0,1]. The numbers of time steps of t and τ equal the length of the input sequence and a user-defined number (N_{step}) for the reparameterized sequence, respectively. Line 2 computes $\mathfrak{g}(\tau)$ using (2). Both Lines 4 and 6 are the core computations of the algorithm, i.e., (4). Here, the trapezoidal rule is applied for numerical integration

[2]For compactness of notation, n is used instead of N_{step} in mathematical expressions.

Algorithm 1: GORA for SE(3) Sequence.

Inputs : $g(\tau)$: Input SE(3) sequence
Parameter: N_{step}: Number of time steps
Outputs : $g(\tau^*)$: Optimal reparameterized sequence
$\tau^*(t)$: Optimal temporal parameter

1 Initialize τ and t as uniform sampling within $[0, 1]$;
2 Compute $\mathfrak{g}(\tau)$ from (2);
3 **for** *time step* i **do**
4 $\quad F(\tau_i)$ = Integration($\mathfrak{g}^{\frac{1}{2}}(\tau_i)$, $[0, \tau_i]$);
5 **end**
6 Normalize $F(\tau^*) = \frac{F(\tau)}{F(1)}$;
7 Invert $F(\tau^*)$ into $\tau^*(t)$ = Interpolation($F(\tau^*)$, t);
8 Reparameterize as $g(\tau^*)$ = Interpolation(τ^*, $g(\tau)$);

because of its simplicity. $F(\tau)$ is then normalized by dividing the last element, i.e., $F(1)$. Line 7 obtains the global minimizer $\tau^*(t)$ by inverting $F(\tau^*)$, which is numerically conducted by interpolating t with respect to $F(\tau^*)$. Finally, the input data sequence is reparameterized by τ^* by numerical interpolation in SE(3) in Line 8.

C. Computation of Relative Pose Distribution

For a set of poses $\{g_i^{(k)}\}$ in SE(3) at each step i, the sample mean $\mu_i \in \text{SE}(3)$, by definition, satisfies

$$\sum_{k=1}^{m} \log(\mu_i^{-1} g_i^{(k)}) = \mathbb{O} \tag{5}$$

which can be iteratively solved [54]. Here juxtaposition of two elements of SE(3) implies multiplication as homogeneous transformation matrices and $\log(\cdot)$ is the matrix logarithm. The mean trajectory can be directly computed from the demonstration set.

The initial covariance $\Sigma_{i,i+1}$ encodes the uncertainty of $(i+1)$th step given the ith step. It is estimated by the set of relative poses, i.e., $\{\Delta_{i,i+1}^{(k)} = (g_i^{(k)})^{-1} g_{i+1}^{(k)}\}$. With this set, the sample covariance can be computed as

$$\Sigma_{i,i+1} = \frac{1}{m} \sum_{k=1}^{m} \log^{\vee}\left(\mu_{i,i+1}^{-1} \Delta_{i,i+1}^{(k)}\right) \log^{\vee T}\left(\mu_{i,i+1}^{-1} \Delta_{i,i+1}^{(k)}\right) \tag{6}$$

where $\mu_{i,i+1}$ can be computed using (5) but with the relative poses $\Delta_{i,i+1}^{(k)}$ as inputs, and the $\vee$ operator extracts the Lie algebra coefficients into a vector (as defined in [55]).

D. Probabilistic Encoding of Joint Distribution on SE(3) Trajectories

After computing the trajectory distribution with mean $\{\mu_0, \mu_1, \ldots, \mu_n\}$, $\mu_i \in \text{SE}(3)$ and covariance between adjacent steps $\{\Sigma_{0,1}, \Sigma_{1,2}, \ldots, \Sigma_{n-1,n}\}$, $\Sigma_{i,i+1} \in \mathbb{R}^{6\times 6}$ from Section II-I-C, the joint distributions of the whole trajectory can be computed. The idea is to learn the distribution locally with end constraints. The solution is inspired by the concept of *loop entropy* [11], in which the probabilistic description of ensembles of end-constrained SE(3) paths was formulated. Assuming the variation of ith pose only depends on its two neighboring poses and $g_0 = \mu_0$ is fixed, the joint probability density can be expressed as

$$\rho(g_1, g_2, \ldots, g_n) = \prod_{i=0}^{n-1} \rho(g_{i+1}|g_i) \tag{7}$$

where $n = N_{\text{step}}$, and $\rho(g_{i+1}|g_i)$ is the conditional probability of the $(i+1)$th pose given the ith pose.

If the intermediate steps along the trajectory are subject to Gaussian distributions with small variations, explicit results can be shown as

$$\rho(g_1, g_2, \ldots, g_n) = \eta \exp\left(-\frac{1}{2} \boldsymbol{x}_{1,\ldots,n}^T \Sigma_{1,\ldots,n}^{-1} \boldsymbol{x}_{1,\ldots,n}\right) \tag{8}$$

where $\exp(\cdot)$ here is the scalar exponential, $\eta = (2\pi)^{-3(n-1)} |\det \Sigma_{1,\ldots,n}|^{-\frac{1}{2}}$ is the normalizing constant and

$$\boldsymbol{x}_{1,\ldots,n} \doteq [\boldsymbol{x}_1^T, \ldots, \boldsymbol{x}_i^T, \ldots, \boldsymbol{x}_n^T]^T \in \mathbb{R}^{6n}$$

where $\boldsymbol{x}_i = \log^{\vee}(\mu_i^{-1} g_i)$. $\Sigma_{1,\ldots,n}$ is a block matrix, with each block being in size 6×6. The (i,i)th block is expressed as $\Sigma_{1,\ldots,n}(i,i)$ (the same representation for its inverse matrix $\Sigma_{1,\ldots,n}^{-1}$). The nonzero blocks of $\Sigma_{1,\ldots,n}^{-1}$ are

$$\begin{aligned}
\Sigma_{1,\ldots,n}^{-1}(i,i) &= \begin{cases} \Sigma_{i-1,i}^{-1} + \widetilde{\Sigma}_{i,i+1}^{-1} & (i \neq n) \\ \Sigma_{i-1,i}^{-1} & (i = n) \end{cases} \\
\Sigma_{1,\ldots,n}^{-1}(i,i+1) &= -\text{Ad}_{i,i+1}^{-T} \Sigma_{i,i+1}^{-1} \ (i \neq n) \\
\Sigma_{1,\ldots,n}^{-1}(i+1,i) &= -\Sigma_{i,i+1}^{-1} \text{Ad}_{i,i+1}^{-1} \ (i \neq n)
\end{aligned} \tag{9}$$

where $\Sigma_{1,\ldots,n} \in \mathbb{R}^{6n\times 6n}$. Ad is explained as follows. Let $\text{Ad}(g)$ denote the adjoint operator for g in a Lie group, which is defined as

$$\text{Ad}(g)\widehat{\boldsymbol{x}} \doteq \frac{d}{dt}\left(g e^{t\widehat{\boldsymbol{x}}} g^{-1}\right)\Big|_{t=0} = g\widehat{\boldsymbol{x}}g^{-1}$$

where $\widehat{\cdot}$ is the hat map, which is the inverse operation of $\vee$ that maps from a vector space into the Lie algebra. The adjoint operator maps from the Lie group to the set of all invertible linear transformations of the Lie algebra onto itself [55]. Note that $\text{Ad}(g)$ is an operator, but can be expressed as a matrix, denoted as Ad_g. By direct calculations, $\text{Ad}_g \boldsymbol{x} = (g\widehat{\boldsymbol{x}}g^{-1})^{\vee}$. Specifically for SE(3), the adjoint matrix can be computed as

$$\text{Ad}_g \doteq \begin{pmatrix} R & \mathbb{O}_{3\times 3} \\ \widehat{\mathbf{t}}R & R \end{pmatrix} \in \mathbb{R}^{6\times 6} \tag{10}$$

where R and $\mathbf{t}$ are the rotation and translation parts of g, respectively. Here, $\widehat{\cdot}$ is the hat operator in $\mathfrak{so}(3)$, which converts the vector $\mathbf{t}$ into a skew-symmetric matrix [55]. For the sake of simplicity, the adjoint matrix for the relative mean poses between the adjacent time steps in (9), i.e., μ_i and μ_{i+1}, is defined as

$$\text{Ad}_{i,i+1} \doteq \text{Ad}_{\mu_i^{-1}\mu_{i+1}}.$$

The covariance between the relative time steps can then be computed as

$$\widetilde{\Sigma}_{i,i+1} = \text{Ad}_{i,i+1} \Sigma_{i,i+1} \text{Ad}_{i,i+1}^T. \tag{11}$$

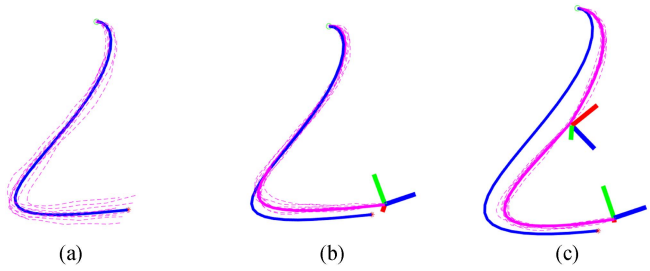


Fig. 2. Examples of the adaptation to new via points with uncertainties. The solid blue and magenta curves are the means of the encoded joint prior (9) and posterior (14) distribution, respectively. Dashed magenta curves are the random trajectory samples from the probability distribution. (a) Original trajectory mean and samples. (b) Adaptation to a new goal pose. (c) Adaptation to new via points.

The inverse operations in (9) are simply matrix inverses. In this formulation, the variable $\boldsymbol{x}_{1,\dots,n} \sim \mathcal{N}(\mathbf{0}, \Sigma_{1,\dots,n})$ is subject to a Gaussian distribution with zero mean. Detailed derivations can be referred to Appendix A1.

E. Adaptation to Novel Situations

One of the most essential abilities of an LfD method is its adaptability to novel situations. The adaptation to via points is first discussed. Then, its equivariant property is studied when there is a change of viewing frame. Finally, the learned distribution is fused with the robot-specific workspace density.

1) Adaptation to Via Points With Uncertainties: Suppose that the robot is asked to pass a via point $g_i^* \in \mathrm{SE}(3)$ with uncertainty, which is described by covariance matrix Σ_i^*. The posterior distribution can be computed as follows. Using an observation model, the new via point be defined as a function of the variable $\boldsymbol{x}_i$, i.e.,

$$g_i^* \doteq \mu_i \exp\left(\widehat{\boldsymbol{x}}_i\right) \exp\left(\widehat{\boldsymbol{\xi}}\right) = \mu_i \exp\left(\widehat{C_i \boldsymbol{x}}_{1,\dots,n}\right) \exp\left(\widehat{\boldsymbol{\xi}}\right) \tag{12}$$

where $\exp(\widehat{\cdot})$ here is the matrix exponential that maps from Lie algebra to Lie group, and $\exp(\widehat{\boldsymbol{\xi}}) \sim \mathcal{N}(\mathbb{I}, \Sigma_i^*)$ is subject to the desired covariance. Here

$$C_i = \mathbf{e}_i^T \otimes \mathbb{I}_6 \in \mathbb{R}^{6 \times 6n}$$

selects the block of variables for the ith step, $\otimes$ denotes the Kronecker product, $\mathbf{e}_i \in \mathbb{R}^n$ is the n-dimensional basis vector and $\mathbb{I}_6$ is a 6-D identity matrix. Defining a new variable, i.e., $\boldsymbol{y} \doteq \log^\vee(\mu_i^{-1} g_i^*)$, gives

$$\boldsymbol{y} = \log^\vee\left(\exp\left(\widehat{C_i \boldsymbol{x}}_{1,\dots,n}\right) \exp\left(\widehat{\boldsymbol{\xi}}\right)\right) \approx C_i \boldsymbol{x}_{1,\dots,n} + \boldsymbol{\xi}\,. \tag{13}$$

Using (13), the mean and covariance of the posterior distribution can be computed as

$$\begin{aligned} K_i &\doteq \Sigma_{1,\dots,n} C_i^T \left(C_i \Sigma_{1,\dots,n} C_i^T + \Sigma_i^*\right)^{-1} \in \mathbb{R}^{6n \times 6} \\ \bar{\boldsymbol{x}}_{1,\dots,n} &= K_i \log^\vee\left(\mu_i^{-1} g_i^*\right) \\ \bar{\Sigma}_{1,\dots,n} &= (\mathbb{I}_{6n} - K_i C_i) \Sigma_{1,\dots,n} \end{aligned} \tag{14}$$

where $\bar{\boldsymbol{x}}_{1,\dots,n}$ and $\bar{\Sigma}_{1,\dots,n}$ are the posterior estimations of the mean variable and covariance, respectively. Fig. 2 shows the adaptation to novel uncertain via points.

2) Equivariant Adaptation to the Change of View: To change the viewing frame, a group action is applied. Suppose $h \in \mathrm{SE}(3)$ is the relative transformation from the current frame (O) to a new frame (A), then the pose g viewed in frame O can be switched to be viewed in frame A as $g^o = h^{-1} g h$ [50].

The conditional probability between two adjacent frames after the change of view can be computed as

$$\boldsymbol{x}_i^o = \log^\vee(h^{-1} \mu_i^{-1} g_i h) = \mathrm{Ad}_h^{-1} \boldsymbol{x}_i\,. \tag{15}$$

The same expression can be obtained for $\boldsymbol{x}_{i+1}^o$. Then, the joint variable $\boldsymbol{x}_{i,i+1} = [\boldsymbol{x}_i^T, \boldsymbol{x}_{i+1}^T]^T \in \mathbb{R}^{12}$ has the expression after the change of view as

$$\boldsymbol{x}_{i,i+1}^o = \begin{pmatrix} \mathrm{Ad}_h^{-1} & \mathbb{O}_{6\times 6} \\ \mathbb{O}_{6\times 6} & \mathrm{Ad}_h^{-1} \end{pmatrix} \boldsymbol{x}_{i,i+1}\,. \tag{16}$$

For a Gaussian distribution, explicitly writing down the quadratic term in the exponent gives

$$\rho(g_{i+1}|g_i) \propto \exp\left(-\frac{1}{2} \left\| \boldsymbol{x}_{i+1}^o - \mathrm{Ad}_{i,i+1}^o \boldsymbol{x}_i^o \right\|^2_{\Sigma_{i,i+1}^{o\,-1}}\right) \tag{17}$$

where $\Sigma_{i,i+1}^o = \mathrm{Ad}_h^{-1} \Sigma_{i,i+1} \mathrm{Ad}_h^{-T}$, $\mathrm{Ad}_{i,i+1}^o \doteq \mathrm{Ad}_{\mu_i^{o\,-1} \mu_{i+1}^o}$, $\|\boldsymbol{x}\|_W^2 = \boldsymbol{x}^T W \boldsymbol{x}$ for a vector $\boldsymbol{x} \in \mathbb{R}^n$ and $-T$ denotes the inverse of the transpose of a matrix. Detailed derivation can be found in Appendix A2. From (17), the conditional probability viewed in frame A is also a Gaussian with zero mean and covariance $\Sigma_{i,i+1}^o$. Therefore, for the joint variable $\boldsymbol{x}_{i,i+1}$, we can compute the inverse of joint covariance in the new frame as

$$\Sigma_{i,i+1}^{o\,-1} = \begin{pmatrix} \mathrm{Ad}_{i,i+1}^{o\,-T} \Sigma_{i,i+1}^{o\,-1} \mathrm{Ad}_{i,i+1}^{o\,-1} & -\mathrm{Ad}_{i,i+1}^{o\,-T} \Sigma_{i,i+1}^{o\,-1} \\ -\Sigma_{i,i+1}^{o\,-1} \mathrm{Ad}_{i,i+1}^{o\,-1} & \Sigma_{i,i+1}^{o\,-1} \end{pmatrix}. \tag{18}$$

From the above derivations we can write

$$f(h \circ \boldsymbol{x}_{i,i+1}) = h \odot f(\boldsymbol{x}_{i,i+1}) \tag{19}$$

where $h \circ \boldsymbol{x}_{i,i+1}$ has the form in (16) and $h \odot f(\boldsymbol{x}_{i,i+1}) \sim \mathcal{N}(\mathbf{0}, \Sigma_{i,i+1}^o)$ with the covariance from (18). This shows the equivariance of the conditional distribution under the change of view.

Using the same derivation process, the distribution of the whole trajectory can be obtained. The variable becomes

$$\boldsymbol{x}_{1,\dots,n}^o = \left(\mathbb{I}_n \otimes \mathrm{Ad}_h^{-1}\right) \boldsymbol{x}_{1,\dots,n} \tag{20}$$

where $\boldsymbol{x}_{1,\dots,n}$ is defined in (8). The distribution has zero mean and covariance, which has the same structure as (9). The difference is for (11), which becomes

$$\begin{aligned} \Sigma_{i,i+1}^o &= \mathrm{Ad}_h^{-1} \Sigma_{i,i+1} \mathrm{Ad}_h^{-T}\,, \text{and} \\ \widetilde{\Sigma}_{i,i+1}^o &= \mathrm{Ad}_{i,i+1}^o \Sigma_{i,i+1}^o \mathrm{Ad}_{i,i+1}^{o\,T}. \end{aligned} \tag{21}$$

Equations (20) and (21) also satisfy the equivariance property [as in (19)]. Demonstrations on the equivariance property for the change of viewing frame are shown in Fig. 3.

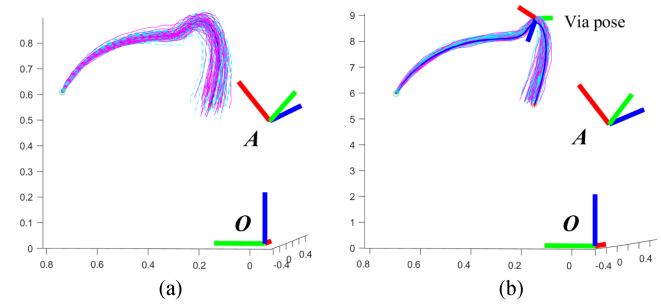


Fig. 3. Equivariance property under the change of viewing frame. The samples as viewed in the original frame O are shown in magenta, and samples after the change of view using (21) are shown in cyan (trajectories are sampled as viewed in the new frame A but transformed back to frame O for each pose for illustration purpose). (a) Encoded joint distribution. (b) Distribution after conditioning on a via point.

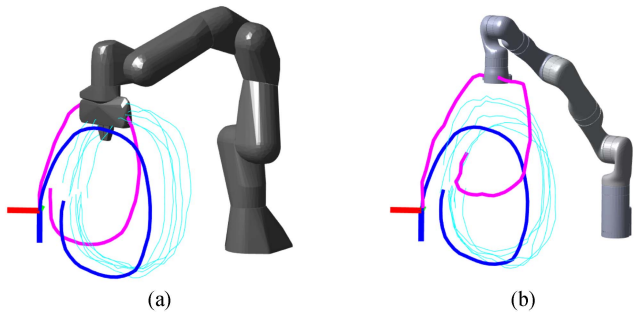


Fig. 4. Fusion with robot-specific workspace density. Thin cyan curves are the demonstrated trajectories using the Franka robot; solid thick blue and magenta curves are the mean trajectory without and with the fusion, respectively. The end effector is placed at a random intermediate step along the fused trajectory mean. (a) Franka Emika Panda. (b) Kinova Gen3.

3) Adaptation to Robot-Specific Workspace Density: PRIMP learns motions in the workspace of the robot instead of joint space, which makes skill transfer among different robots easy. In other words, the demonstrations are conducted using one robot, and the learned workspace distribution can be used by another robot with a different kinematic structure. However, an important issue to consider is the adaptation to robot-specific workspace limits and reachability. Previous work has extensively investigated the density of the robot workspaces, in which the more reachable space of the end effector has a higher probability [10], [56]. The workspace density was initially constructed by discretizing the workspace and computing the histogram of the poses at each voxel [57], [58]. To make computations more efficient, Gaussian distributions were further used to approximate the workspace density of manipulators [59]. This approach has been used to efficiently solve the inverse kinematics of a serial hyperredundant manipulator. In this work, the same concept is applied but in a different way. The workspace density is used to inform the learned trajectory distribution, in order to stay closer to the higher probability region within the robot workspace. The following introduces how to fuse the learned distribution with the workspace density of a specific robot, so as to maximize the mobility when moving along the learned trajectory.

The mathematical foundation is based on the convolution on $\mathrm{SE}(3)$, i.e.,

$$(\rho_1 * \rho_2)(g) \doteq \int_G \rho_1(h)\rho_2(h^{-1}g)dh$$

where h is an integration variable and dh is the Haar measure on $\mathrm{SE}(3)$ [10]. For a serial manipulator with m joints, the workspace density is computed as the convolution of Gaussian distributions, i.e., $\rho_{\mathrm{wd}}(g) = (\rho_1 * \cdots * \rho_m)(g)$, where $\rho_j(g), j = 1, \ldots, m$ is the Gaussian distribution of all possible poses of the distal end of the jth link. In practice, a discrete set of joint angles is sampled uniformly. The pose of the distal end is computed and approximated as a Gaussian defined by sample mean and covariance using (5) and (6). The resulting workspace density is also a Gaussian in $\mathrm{SE}(3)$, whose mean can be computed as $g_{\mathrm{wd}} = \mu_1\, \mu_2 \ldots \mu_m$. The covariance Σ_{wd} can be approximated by an iterative process of twofold convolution, i.e., $\Sigma_{1*2} = \mathrm{Ad}_{\mu_2}^{-1}\, \Sigma_1\, \mathrm{Ad}_{\mu_2}^{-T} + \Sigma_2$, to the first order [56], [60], [61].

Then, the distribution of each intermediate pose along the trajectory is conditioned by this density function. The idea is analogous to that in Section III-E1, in which each intermediate pose is asked to pass through the desired g_{wd} with uncertainty Σ_{wd}. For each step i, the workspace density can be approximated as

$$g_{\mathrm{wd}} \doteq \mu_i \exp\left(\widehat{\boldsymbol{x}}_i\right) \exp\left(\widehat{\boldsymbol{\xi}}\right) \tag{22}$$

where $\exp(\widehat{\boldsymbol{\xi}}) \sim \mathcal{N}(\mathbb{I}, \Sigma_{\mathrm{wd}})$. Let $\boldsymbol{y}_i \doteq \log^{\vee}(\mu_i^{-1} g_{\mathrm{wd}})$, then

$$\boldsymbol{y}_i \approx \boldsymbol{x}_i + \boldsymbol{\xi}\,.$$

Stacking the variables for all steps together gives $\boldsymbol{y} = [\boldsymbol{y}_1^T, \ldots, \boldsymbol{y}_n^T]^T$. Therefore, the trajectory distribution after fusing with the workspace density can be computed by

$$\begin{aligned} K &\doteq \Sigma_{1,\ldots,n}\left(\Sigma_{1,\ldots,n} + \mathbb{I}_n \otimes \Sigma_{\mathrm{wd}}\right)^{-1} \\ \bar{\boldsymbol{x}}_{1,\ldots,n} &= K\boldsymbol{y} \\ \bar{\Sigma}_{1,\ldots,n} &= (\mathbb{I}_{6n} - K)\Sigma_{1,\ldots,n}\,. \end{aligned} \tag{23}$$

Fig. 4 demonstrates the fusion with robot-specific workspace density. The demonstrations are conducted using the Franka Emika Panda robot. Then, PRIMP is fused by the workspace density of both Panda and Kinova Gen3 robots, followed by adapting to a via point.

F. Extensions to the PCG

In addition to $\mathrm{SE}(3)$, the trajectory can also be represented by $\mathrm{PCG}(3)$, [50]. An element in $\mathrm{PCG}(3)$ is represented as a rotation and translation pair, i.e.,

$$g \doteq (R, \mathbf{t}) \in \mathrm{PCG}(3) = \mathrm{SO}(3) \times \mathbb{R}^3$$

which is a direct product (as compared with the semidirect product for $\mathrm{SE}(3)$). The exponential mapping for $\mathrm{PCG}(3)$ is defined as $\exp(\boldsymbol{\xi}) \doteq (\exp(\widehat{\boldsymbol{\omega}}), \mathbf{t})$, where $\boldsymbol{\xi} \doteq [\boldsymbol{\omega}^T, \mathbf{t}^T]^T \in \mathbb{R}^6$ and $\widehat{\boldsymbol{\omega}} = \log(R) \in \mathfrak{so}(3)$. The logarithm mapping is then defined as $\log(g) \doteq (\log(R), \mathbf{t}) \in \mathfrak{pcg}(3)$ and $\log^{\vee}(g) = [(\log^{\vee}(R))^T, \mathbf{t}^T]^T = \boldsymbol{\xi} \in \mathbb{R}^6$. Then, the relative pose between

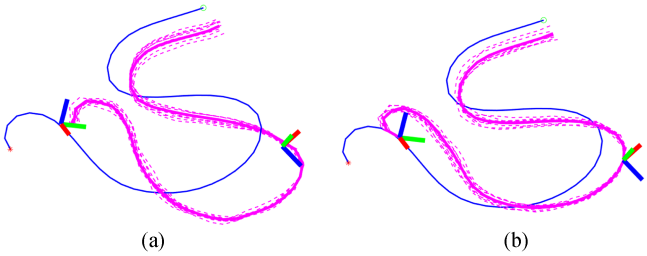


Fig. 5. PRIMP modeled on SE(3) and PCG(3). Solid blue and magenta curves are the mean of the demonstrated and conditioned trajectories, respectively. (a) SE(3). (b) PCG(3).

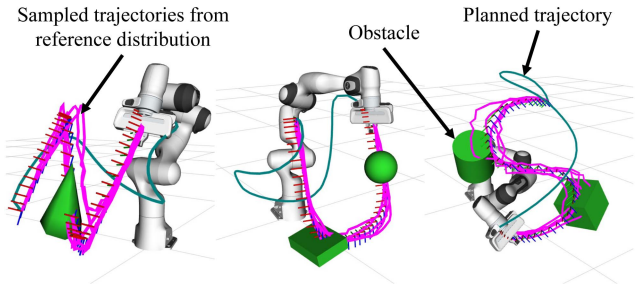


Fig. 6. Motion planning using Workspace-STOMP to follow the learned trajectory distribution from PRIMP.

g_1 and g_2 becomes $\Delta_{1,2} \doteq (R_1^T R_2, \mathbf{t}_2 - \mathbf{t}_1)$. The adjoint matrix of PCG(3) can be computed as

$$\mathrm{Ad}_g \doteq \begin{pmatrix} R & \mathbb{O}_{3\times3} \\ \mathbb{O}_{3\times3} & \mathbb{I}_3 \end{pmatrix} \in \mathbb{R}^{6\times6}. \tag{24}$$

To compute the trajectory distribution in (8), the local variable for each frame becomes $\boldsymbol{x}_i \doteq \begin{pmatrix} \log^\vee(Q_i^T R_i) \\ \mathbf{t}_i - \mathbf{s}_i \end{pmatrix}$, where Q_i and s_i are the means of rotation and translation part, respectively. Using the new local variable and PCG(3) operations, (14), (5), and (6) are changed accordingly. One of the advantages of using PCG(3) is that the orientation and position of a pose are decoupled, resulting in separate computations when concatenating adjacent time steps. As for interpolating among discrete poses, the total translational distance is, in general, shorter than the SE(3) formulation. A qualitative comparison for the SE(3) and PCG(3) trajectories is demonstrated in Fig. 5.

IV. Motion Planning Guided by PRIMP

This section introduces *Workspace-STOMP*, a novel guided motion planning algorithm using the trajectory distribution computed by PRIMP. A cost function for the end effector trajectory is proposed to guide the STOMP algorithm. The cost is computed based on the distance metric in SE(3) between each rollout trajectory at each iteration and the workspace trajectory distribution learned by PRIMP.

At each iteration, STOMP defines a set of random samples in joint space, each of which is denoted as a "rollout," i.e., $\mathbf{q}$. To compute the distance metric of each rollout with the learned trajectory distribution, the trajectory of the end effector is computed via forward kinematics, denoted as $g(\mathbf{q}, t) \in \mathrm{SE}(3) \times \mathcal{T}$. Then, a number of m_r random samples from the reference trajectory distribution are generated, denoted as $g_r^{(k)} = (R_r^{(k)}, \mathbf{t}_r^{(k)})$. The cost function $\mathbf{c}(\mathbf{q}_i, t_i)$ for ith time step is computed as

$$\mathbf{c}(\mathbf{q}_i, t_i) = \frac{1}{m_\mathrm{r}} \sum_{k=1}^{m_\mathrm{r}} \Big(w_\mathrm{rot} \left\| \log^\vee \left(R^T(\mathbf{q}_i, t_i)\, R_\mathrm{r}^{(k)}(t_i) \right) \right\| + w_\mathrm{tran} \left\| \mathbf{t}(\mathbf{q}_i, t_i) - \mathbf{t}_\mathrm{r}^{(k)}(t_i) \right\| \Big). \tag{25}$$

The weights w_rot and w_tran for rotation and translation parts in the distance function are set by users. In fact, other distances (e.g., Mahalanobis distance or log-likelihood) can also be applied here, in order to measure the proximity from the rollout to the reference trajectory distribution. However, when using metrics that include a covariance matrix, the cost value will be hard to scale to balance with other costs defined in STOMP. For example, if the distribution is concentrated (covariance is very small), the Mahalanobis distance will be very large even when the rollout is just a bit far from the mean. On the other hand, when computing the distance with samples, it is not sensitive to the magnitudes of the covariance, making it easier to scale the units of the cost value. In the highly concentrated case, the distance value will converge to the geodesic distance between the rollout and the mean reference trajectory. The computational process is shown in Algorithm 2.

Algorithm 2: Cost Function for Workspace-STOMP based on Trajectory Distribution.

Inputs : $\mathbf{q}$: Rollout joint angles; $\left\{g_\mathrm{r}^{(k)}(t)\right\}$: Sampled trajectories from the distribution learned by PRIMP
Parameters: w_rot, w_tran: Weights for rotation and translation parts in the distance function
Outputs : $\mathbf{c}(\mathbf{q}, t)$: Cost value for each rollout

```
for time step i do
    g(q_i, t_i) = ForwardKinematics(q_i);
    Compute c(q_i, t_i) using (25) ;
end
```

The planner is initialized by the mean trajectory of the learned distribution. Then, a plug-in package of the proposed cost function is implemented in MoveIt! platform [62]. Simulations of writing letters "N," "U," and "S" using the proposed Workspace-STOMP are shown in Fig. 6.

V. Evaluations of the Proposed Method

The evaluations of the proposed PRIMP and Workspace-STOMP algorithms are conducted in this section: 1) PRIMP on SE(3) and PCG(3) as well as some popular state-of-the-art LfD methods (see Section V-B); 2) properties of PRIMP in extrapolation and single-demonstration cases (see Section V-D); and 3) variants of STOMP for motion planning (see Section V-E).

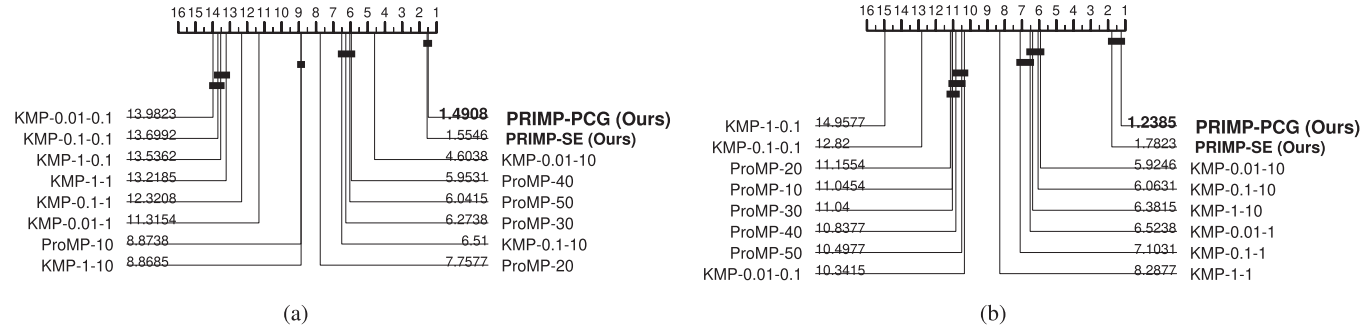


Fig. 7. Critical difference diagram for performance ranking for the 26 categories in the LASA handwriting dataset, with $\alpha = 0.05$. The numbers are the averaged ranking of the methods, the smaller (further to the right) the better. The methods connected by bold lines do not have significant differences in performance. Different sets of parameters for ProMP and KMP are labeled after the method name. For ProMP, the format is "ProMP-[number of basis functions]"; and for KMP, the format is "KMP-[scale factor of regularization]-[width of Gaussian for each kernel]." (a) D_{demo}, translation part. (b) D_{via}, translation part.

A. Comparison Metrics

1) Similarity With Original Trajectory: The averaged distance is computed between samples from the learned distribution and the demonstrated trajectories. Mathematically, the metric is defined as

$$D_{\text{demo}} \doteq \frac{1}{m\,s}\sum_{k=1}^{m}\sum_{j=1}^{s}\text{DTW}\left(\{g^{(j)}_{\text{learned}}\},\{g^{(k)}_{\text{demo}}\}\right) \tag{26}$$

where m, s are the numbers of originally demonstrated trajectories and sampled trajectories, respectively. $\text{DTW}(\{\cdot\},\{\cdot\})$ denotes the DTW distance between two sequences, where the distance metric in Lie group is used. In the benchmark experiments, the metrics for the rotation and translation parts are computed separately. In particular, $d_{\text{rot}}(R_1,R_2) = \|R_1 - R_2\|_F$ ($\|\cdot\|_F$ denotes the Frobenius norm) and $d_{\text{tran}}(\mathbf{t}_1,\mathbf{t}_2) = \|\mathbf{t}_1 - \mathbf{t}_2\|_2$ ($\|\cdot\|_2$ denotes the 2-norm).

2) Adaptability to New Via Point: The distance between the computed pose at a specific time step i and the designated pose mean at the same step, i.e., μ^*_i, is computed as

$$D_{\text{via}} \doteq \frac{1}{s}\sum_{j=1}^{s} d\left(g^{(j)}_i,\mu^*_i\right). \tag{27}$$

3) Similarity With Reference Trajectory: The similarity of the computed trajectory with the reference is evaluated after motion planning. Given the mean of the reference end-effector trajectories $\{g_{\text{ref,i}} = (R_{\text{ref,i}},\mathbf{t}_{\text{ref,i}})\}$ and planned trajectory $\{g_{\text{plan,i}} = (R_{\text{plan,i}},\mathbf{t}_{\text{plan,i}})\}$, the error metric is defined as the accumulated distance between each step along the two trajectories, i.e.,

$$\begin{aligned} e_{\text{rot}} &= \sum_{i=1}^{n}\|\log^{\vee}\left(R^T_{\text{ref,i}}R_{\text{plan,i}}\right)\|_2 \\ e_{\text{tran}} &= \sum_{i=1}^{n}\|\mathbf{t}_{\text{ref,i}} - \mathbf{t}_{\text{plan,i}}\|_2\,. \end{aligned} \tag{28}$$

B. Benchmarks Among LfD Methods

The data for LfD benchmarks include three types, which are as follows:

1) LASA hand-writing dataset [63];
2) end effector positions generated in simulation;
3) real-world kinesthetic teaching for several common daily tasks.

The former two types only include positional trajectories while the third type involves both position and orientation data. The existing probabilistic methods to be compared include 1) ProMP [33], which only deals with the translation part of the trajectories; and 2) KMP [7], [38], which considers both translation and orientation parts.

1) Parameters for the Baselines: Different sets of parameters for the two baseline methods are considered. For ProMP, radial basis functions are chosen to encode the latent-space representation, where the number of Gaussian functions is the parameter to be tuned. For the KMP method, the width of Gaussian for each kernel and the scale factor of the regularization term play essential roles in encoding and generalizing the learned trajectory distribution. For both methods, we are using the open-sourced implementations.[3]

2) LASA Handwriting Dataset: This dataset involves 2-D handwriting demonstrations. We then augment by adding a zero z-coordinate and identity 3-D orientation for each pose in the trajectory. The 4 sets consisting of multimodal data are omitted, so the comparisons are conducted in the remaining 26 datasets. The distance metrics in (26) and (27) are used to evaluate the performance. Since only translation is included in the demonstration, the evaluations only include the translation part. All the compared methods are ranked among each of the 26 categories. The averaged rankings are shown in a critical difference diagram in Fig. 7.

3) Demonstrations in Simulation: The simulations are conducted by virtually dragging the end effector to draw symbols and letters in MoveIt. The orientation of the end effector is fixed. Fig. 8 shows the benchmark results for the similarity to the demonstrated trajectories [see Fig. 8(a)–(c)] and adaptability to via points [see Fig. 8(d)–(f)].

[3]Implementations are sourced for ProMP from https://github.com/dfki-ric/movement_primitives/ and for KMP from https://github.com/yanlongtu/robInfLib-matlab/.

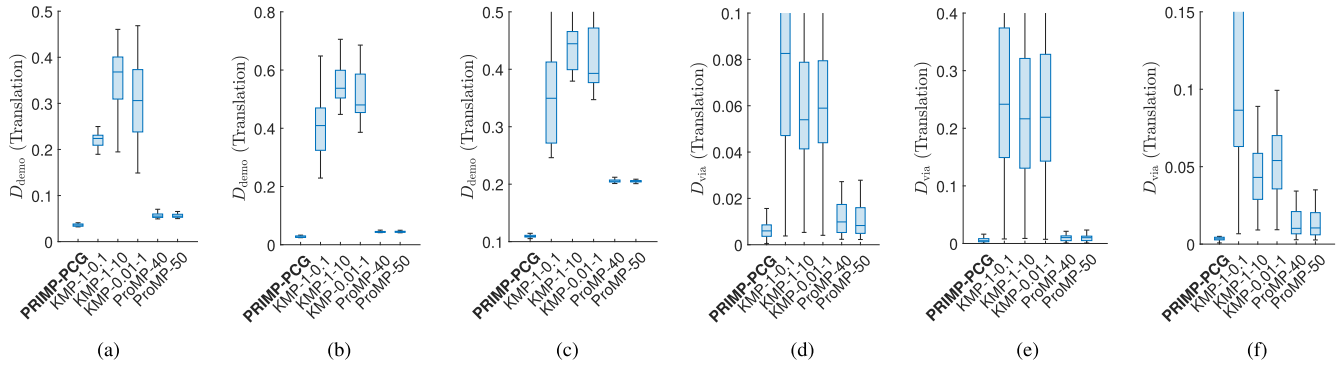
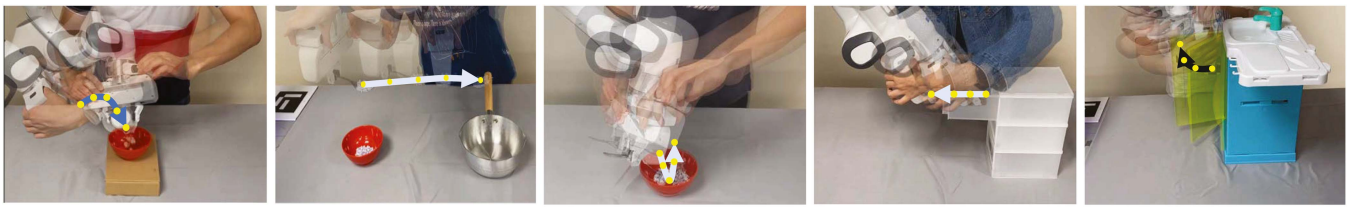


Fig. 8. Benchmark results for LfD methods in simulated data. (a) D_{demo}, "Letter N." (b) D_{demo}, "letter U." (c) D_{demo}, "Letter S." (d) D_{via}, "Letter N." (e) D_{via}, "Letter U." (f) D_{via}, "Letter S."

TABLE I
DAILY OBJECTS AND COMMON TASKS USED IN THE BENCHMARK

Task ID	Task name	Object	Tool	Motion primitives	Task description
1	Pouring	Container	Cup	Pouring	Pour particles to a container
2	Transporting	–	Spoon	Transporting	Transport particles without spillage
3	Scooping	Container	Spoon	Scooping	Scoop particles from a container
4	Drawer Opening	Drawer	Handle	Opening	Open the sliding door of a drawer
5	Door Opening	Door	Handle	Opening	Open the door with hinge



Task 1: Pouring | Task 2: Transporting | Task 3: Scooping | Task 4: Drawer Opening | Task 5: Door Opening

Fig. 9. Kinesthetic demonstrations for different tasks.

4) Physical Demonstrations on Real-World Tasks: We consider common tasks in daily household environments, which are summarized in Table I. At first, for each task, the human demonstrator drags the end effector of the Panda robot, each with 5–10 trials (see Fig. 9). Then, with manually defined 50 different pairs of goals and via points, different LfD algorithms are applied to reproduce the task. Distance metrics as in (26) and (27) are used for evaluations, which are shown in Tables II and III, respectively[4]. The values in the tables are the mean among all the 50 random trials. The PCG formulation of PRIMP is shown in the results. The parameters for KMP and ProMP are selected as the relatively best performance among all the tasks. Specifically, for KMP, the parameters are selected as $\lambda = 0.1$ and $l = 1$, where λ and l represent the scale factor of regularization term and width of Gaussian for each kernel, respectively; and for ProMP, the number of basis functions is chosen as 30.

[4]The bold color indicates the best result among all the compared methods. The bold numbers have the same meaning in all the other tables that follow.

TABLE II
BENCHMARK RESULTS OF SIMILARITY TO DEMONSTRATIONS AMONG LFD METHODS IN REAL-WORLD TASKS

Task ID	Rotation		Translation		
	PRIMP	KMP	PRIMP	KMP	ProMP
1	**0.522**	1.45	**0.0448**	0.200	0.103
2	**0.313**	0.669	**0.0406**	0.289	0.0812
3	**0.201**	0.508	**0.0377**	0.278	0.0527
4	**0.275**	0.372	**0.0168**	0.134	0.0293
5	**0.106**	1.61	**0.00852**	2.20	0.0133

From the benchmark results in different datasets, PRIMP outperforms other probabilistic methods being compared in most cases, in terms of similarity metric for the whole trajectory as well as the distance to the desired via point. The computational time for encoding and generalization can achieve the level of milliseconds, with an average of less than 50 ms, which is more than five times faster than the compared counterparts. Ours leads the performance ranking with a critical difference in the LASA

TABLE III
BENCHMARK RESULTS OF ADAPTATION TO VIA POINTS AMONG LFD METHODS IN REAL-WORLD TASKS

Task ID	Rotation		Translation		
	PRIMP	KMP	PRIMP	KMP	ProMP
1	**0.00748**	0.0869	0.00515	0.0626	**0.00438**
2	**0.00775**	0.0753	**0.00407**	0.0802	0.00564
3	**0.00647**	0.0970	**0.00381**	0.0816	0.00432
4	**0.00714**	0.0868	0.00498	0.116	**0.00277**
5	**0.0175**	0.523	**0.00495**	0.369	0.00578

TABLE IV
ABLATION STUDY FOR GORA PREPROCESS

Dataset	Metric	PRIMP	No GORA	With DTW
LASA	D_{demo} (Trans.)	**2.26**	2.69	6.19
Simulation	D_{demo} (Trans.)	**0.084**	0.092	0.15
Real world	D_{demo} (Rot.)	**0.28**	0.31	0.34
Real world	D_{demo} (Trans.)	**0.073**	0.074	0.089
LASA	D_{via} (Trans.)	**0.033**	0.033	0.045
Simulation	D_{via} (Trans.)	**0.034**	0.042	0.092
Real world	D_{via} (Rot.)	**0.025**	0.037	0.052
Real world	D_{via} (Trans.)	**0.037**	0.044	0.049

TABLE V
ABLATION STUDY RESULTS FOR ADAPTATION TO ROBOT-SPECIFIC WORKSPACE DENSITY

Robot	Task ID	Original	Ablated	Improvement
Kinova Gen 3	1	**2.74**	2.82	3.1%
	2	**2.76**	2.91	5.3%
	3	**2.49**	2.95	15.5%
	4	**2.71**	2.95	8.27%
	5	**2.67**	3.05	12.35%
KUKA iiwa 7	1	**2.73**	2.87	4.77%
	2	3.15	**3.09**	−1.89%
	3	**2.15**	2.84	24.06%
	4	**3.09**	3.31	6.46%
	5	**3.30**	3.30	0%
UR 5	1	**2.49**	2.56	2.37%
	2	**2.71**	2.76	1.78%
	3	**2.27**	2.62	13.17%
	4	**2.48**	2.61	5.23%
	5	**2.61**	2.82	7.28%
Atlas (left arm)	1	**3.18**	3.28	2.99%
	2	**2.75**	2.99	8.00%
	3	**3.06**	3.52	13.09%
	4	**2.54**	2.61	2.78%
	5	**2.53**	2.82	10.37%

dataset, as shown in Fig. 7, and outperforms synthetic and real data for daily tasks. For the cases of $\mathrm{SE}(3)$ and $\mathrm{PCG}(3)$, PRIMP performs similarly, with $\mathrm{PCG}(3)$ being slightly better. This is partly because that when translation and orientation parts are separated, the mutual influence or error accumulation between different types of motions is mitigated.

C. Ablation Studies

1) GORA Preprocess for Temporal Alignment: The effect of GORA as a preprocess is evaluated. All the demonstration trajectories are first preprocessed and encoded by PRIMP. Then, with the definition of the desired goal and a via point, the posterior distribution is obtained. The proposed PRIMP method is compared with its ablated version without GORA and the one using DTW as preprocessing. Metrics for similarity with the original trajectory (D_{demo}) and adaptability to new via points (D_{via}) are compared. Table IV shows the comparison results, with metrics for rotation and translation parts being separated. The values in the table are the averaged distance among all data categories in each dataset.

From the results, PRIMP with GORA reproduces the motions with the highest similarity with the demonstrated trajectories and lowest errors to the desired via points in all the cases. It shows the effectiveness of using GORA as a preprocess to align multiple demonstrations in the temporal axis. With the reparameterized trajectories, PRIMP is able to learn the distribution and generalize to novel situations better than not using any preprocess or aligned by the DTW method.

2) Workspace Density Adaptation for Skill Transfers Among Different Robots: The effect of adaptation to robot-specific workspace density is evaluated. Demonstrations are conducted in Franka Panda robot and the robots for execution are changed. Real-world tasks are considered here. The metric is similar to the adaptability to via points as in (27). But, the distance $d(\cdot)$ inside the summation is computed as the Mahalanobis distance from each pose along the trajectory to the workspace density of each robot, i.e.,

$$d\left(g_i^{(j)}, \mu_{\text{wd}}; \Sigma_{\text{wd}}\right) = \left\| \log^{\vee}\left(\mu_{\text{wd}}^{-1} g_i^{(j)}\right) \right\|_{\Sigma_{\text{wd}}^{-1}}. \tag{29}$$

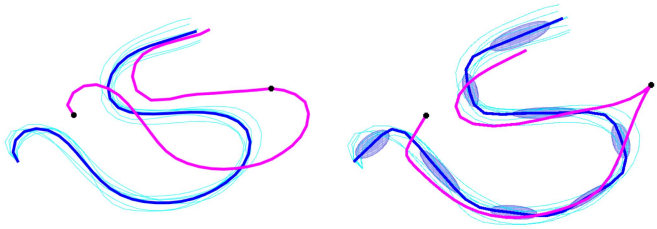
This is because that the Mahalanobis distance measures the proximity of a data point to a distribution, which is more suitable here than simply computing the distance to the mean pose. Table V illustrates the comparisons between PRIMP with and without the adaptation to workspace density of different robots, which are denoted as "Original" and "Ablated," respectively. The percentage of improvements in the closeness to the high probability region is also computed for each robot in each task.

In almost all the tasks with different types of robots, being conditioned by the workspace density pushes the whole learned trajectory closer to the region where the robot has higher probability to reach. The improvements is around 10% on average.

D. Properties of PRIMP

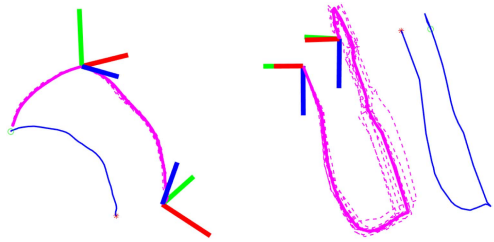
1) Extrapolation: One of the most desirable properties of an LfD method is the ability to adapt to extrapolations when via points are out of the distributions of the demonstrations. A qualitative comparison is shown for the case of extrapolation between PRIMP and KMP methods in Fig. 10 using the LASA handwriting dataset.

The effectiveness of the proposed PRIMP method in the extrapolation case is further shown in quantitative comparisons with KMP in the real-world kinesthetic dataset. The start and goal poses are randomly generated as two via points to be passed through, which are far from the demonstrated trajectories. For PRIMP, the PCG formulation is used; and for KMP, the parameter is chosen as $\lambda = 0.01$ and $l = 10$, which performs relatively the best as compared with other combinations. The similarity



(a) (b)

Fig. 10. Qualitative comparisons for extrapolation between PRIMP and KMP. Positions of via points are indicated by black dots. Solid thick blue and magenta curves are the means of the demonstrated and reproduced trajectories, respectively. For KMP (b), blue ellipsoids indicate the level surface of the GMM. (a) PRIMP. (b) KMP.



(a) (b)

Fig. 11. PRIMP learning from a single demonstration. The blue curve is the only demonstrated trajectory. Solid and dashed magenta curves are the mean and samples from the adapted trajectory after conditioning, respectively. (a) Task 1. (b) Task 4.

TABLE VI
MEAN VALUES OF SIMILARITY AND ADAPTABILITY METRICS FOR THE EXTRAPOLATION CASE IN REAL-WORLD TASKS

Metric	Task ID	Rotation		Translation	
		PRIMP	KMP	PRIMP	KMP
D_{demo}	1	2.27	**2.08**	5.93	**4.91**
	2	**1.82**	2.01	6.28	**5.75**
	3	2.04	**1.86**	4.64	**1.74**
	4	**1.32**	2.18	3.02	**2.11**
	5	**1.71**	1.93	**3.29**	8.53
D_{via}	1	**0.354**	2.16	**1.35**	3.05
	2	**0.667**	1.76	**2.01**	3.77
	3	**0.174**	1.63	**1.03**	2.77
	4	**0.690**	1.88	**3.76**	6.96
	5	**0.973**	2.06	**4.07**	5.07

with demonstrations (D_{demo}) and adaptability to novel situations (D_{via}) that are out of distributions are shown in Table VI. The mean values among 50 random trials are demonstrated, with rotation and translation parts being separated.

Although in some tasks, KMP reproduces the motions more similar to the demonstrations (especially for the translation part), the leads are not significant. On the other hand, ours always has much lower errors to the desired via points, which are far from the demonstrated trajectories. It shows that PRIMP is able to adapt to the novel situations that are extrapolated.

2) Learning From a Single Demonstration: The ability of PRIMP to learn from a single demonstration is illustrated in Fig. 11 qualitatively. The real-world tasks are selected, each of which only consists of one demonstrated trajectory.

Comparisons with learning from the multiple demonstrations are also conducted. The baseline in this part of experiments is defined to learn from all the demonstrated trajectories for the real-world tasks. Tables VII and VIII show the similarity with the baseline (i.e., (26), but using the mean trajectory from the baseline instead of the set of demonstrated trajectories) and the adaptability to via points (27), respectively. The mean values of the metrics are listed with rotation and translation parts separated. In Table VIII, "Single" and "Full" represents the learning from a single demonstration and the full set of demonstrations, respectively.

TABLE VII
MEAN VALUES OF SIMILARITY TO THE BASELINE FOR LEARNING FROM A SINGLE DEMONSTRATION IN REAL-WORLD TASKS

Task ID	Rotation	Translation
1	0.196	0.0271
2	0.0472	0.0225
3	0.150	0.0214
4	0.0133	0.00287
5	0.0445	0.00339

TABLE VIII
MEAN VALUES OF ADAPTABILITY METRIC (D_{via}) FOR LEARNING FROM A SINGLE DEMONSTRATION IN REAL-WORLD TASKS

Task ID	Rotation ($\times 10^{-4}$)		Translation ($\times 10^{-4}$)	
	Single	Full	Single	Full
1	**2.09**	2.10	**1.03**	1.04
2	2.10	**2.08**	**1.05**	1.07
3	2.16	**2.15**	**1.07**	**1.07**
4	2.10	**2.07**	**1.03**	1.04
5	2.06	**2.04**	1.03	**1.02**

Compared to the baseline that uses the full dataset, using a single demonstration can still generate trajectories that are reasonably similar to the demonstration. By manually defining the initial covariance as a diagonal matrix with small values as the entries, PRIMP is still able to adapt to novel via points with small errors. Although the errors are slightly larger than using a full dataset, they are within the same level of magnitude and the differences are not large.

3) Equivariant Adaptation to the Change of View: This set of benchmark experiments demonstrates the ability of PRIMP to adapt to the change of a viewing frame. By using the equivariance property described in Section III-E2, the learned trajectory distribution can be switched in the new local reference frame efficiently. Trajectory samples can be generated locally from the new mean and covariance. On the other hand, after the change of viewing frame, a single trajectory can be transformed by the conjugation. Therefore, the samples in the original frame can also be expressed from the new frame of view. This way of generating samples in the new viewing frame is treated as the baseline of this set of benchmarks.

To evaluate the equivariance property and show its efficiency in generating trajectory samples, the similarity to the samples

TABLE IX
MEAN VALUES OF SIMILARITY TO BASELINE FOR EQUIVARIANT ADAPTATION TO THE CHANGE OF VIEW IN REAL-WORLD TASKS

Task ID	Rotation	Translation
1	0.100	0.0275
2	0.0330	0.0157
3	0.107	0.0247
4	0.0128	0.00783
5	0.0198	0.00732

TABLE X
MEAN VALUES OF ADAPTABILITY (D_{via}) FOR EQUIVARIANT ADAPTATION TO THE CHANGE OF VIEW IN REAL-WORLD TASKS

Task ID	Rotation		Translation	
	Equivariance	Baseline	Equivariance	Baseline
1	0.00564	**0.00537**	0.0125	**0.0103**
2	**0.00580**	0.00595	**0.00603**	0.00826
3	**0.00511**	0.00521	**0.00707**	0.00816
4	0.00647	**0.00624**	0.0101	**0.00920**
5	**0.0163**	0.0164	0.00703	**0.00516**

TABLE XI
MEAN VALUES OF COMPUTATIONAL TIME FOR EQUIVARIANT ADAPTATION TO THE CHANGE OF VIEW IN REAL-WORLD TASKS

Task ID	Equivariance (s)	Baseline (s)
1	**0.0408**	0.109
2	**0.0409**	0.110
3	**0.0404**	0.109
4	**0.0465**	0.123
5	**0.0442**	0.117

generated by the baseline, adaptability to novel via points in the new frame and computational time are compared. In the experiments, 50 random sets of via points are generated, and for each of these trials, a random SE(3) element is generated as the new reference frame. Tables IX and X demonstrate the similarity to the baseline and adaptability to the desired via point after conditioning, respectively. The computational time for generating samples after the change of view is also compared and shown in Table XI.

Using the equivariance property, the trajectory distribution can be transferred to another viewing frame efficiently, i.e., twice faster than the baseline in generating samples after switching the frame. The generated samples are very similar to the baseline and do not have significant differences in terms of the proximity to the desired via points in most cases.

E. Comparisons on Guided Motion Planning

Benchmarks for motion planning are conducted, which include manually defined environments in simulation [as in Fig. 12(a), (c), and (e)]. The real-world tasks, as defined in Section V-B4, are applied here. We compare the proposed Workspace-STOMP planner with (1) STOMP [18] without guidance; and (2) Cartesian-guided STOMP [17]. Note that, the original implementations for both baselines initiate the planner with linearly interpolated trajectories between start and goal. However, to make the comparisons fair, the initial conditions for all planners are set to be the mean of the learned distribution from PRIMP and converted to joint space using inverse kinematics.

TABLE XII
COMPARISONS FOR EXPECTED TASK-SOLVING TIME ($\mathcal{E}_{\text{task}}$, IN SECONDS) AMONG PLANNERS

Scene	Task	**Workspace-STOMP**	STOMP	Cartesian-STOMP
Sparse	1	**0.5122**	1.6200	8.2734
	2	**1.3139**	2.3398	4.0865
	3	**0.6177**	0.6324	28.7984
	4	**0.4601**	1.8083	0.9933
	5	2.2603	**1.6806**	7.9418
Cluttered	1	**1.9570**	2.8787	85.4277
	2	**0.5930**	3.1904	3.0109
	3	**2.2612**	2.9259	∞
	4	0.9289	**0.8243**	1.0676
	5	**0.8002**	1.0744	21.7385
Dense	1	**2.6300**	6.9544	35.7064
	2	1.9358	**1.2964**	2.7239
	3	**54.5261**	∞	∞
	4	**0.4813**	0.6114	0.6362
	5	**0.7046**	0.9586	16.4923

The planning time (T_{plan}) and task success rate (ρ_{task}) are computed. The success of a task is evaluated in either a simulation or a manually defined metric. Specifically, for pouring, transporting, and scooping tasks, the planned trajectories are put into the simulator for execution. For pouring, a cup follows the planned trajectory to pour a set of particles into a bowl. The task is succeeded if the percentage of particles inside the bowl is above 95%. For scooping, a spoon that follows the planned motion scoops particles from the bowl. If any particle appears in the spoon, the task is defined as a success. For the transporting task, a particle is loaded into the spoon. If it is not dropped after executing the planned motion, the task is considered as a success. On the other hand, the drawer and door opening tasks are evaluated by computing the error with the reference trajectories resulting from PRIMP. Each trial is succeeded when the error computed by (28) is less than a threshold (e.g., rotational error is less than 20° and translational error is less than 0.2).

The benchmark results for different tasks in different planning scenes are shown for the planning time in Fig. 12. Each row demonstrates the planning results for one scene, and each column shows one task. Table XII shows the *expected* task-solving time by considering the task success rate [64], i.e.,

$$\mathcal{E}_{\text{plan}} \doteq \frac{\overline{T}_{\text{plan}}}{\rho_{\text{task}}} \tag{30}$$

where $\overline{T}_{\text{plan}}$ is the averaged planning time among all the trials. This metric provides a tradeoff between the planning time and success rate, i.e., a method for a task is desirable if it can be solved in a short time with a high success rate. On the other hand, if a method is able to solve the problem with a slightly higher success rate but has to spend quite a long time, its expected time might still be long. When all the trials fail, the expected solving time is infinity.

In general, our proposed Workspace-STOMP runs much faster and has less deviations among multiple trials than the other guided planner Cartesian-STOMP. In some cases, the vanilla STOMP without guidance runs the fastest since it does not

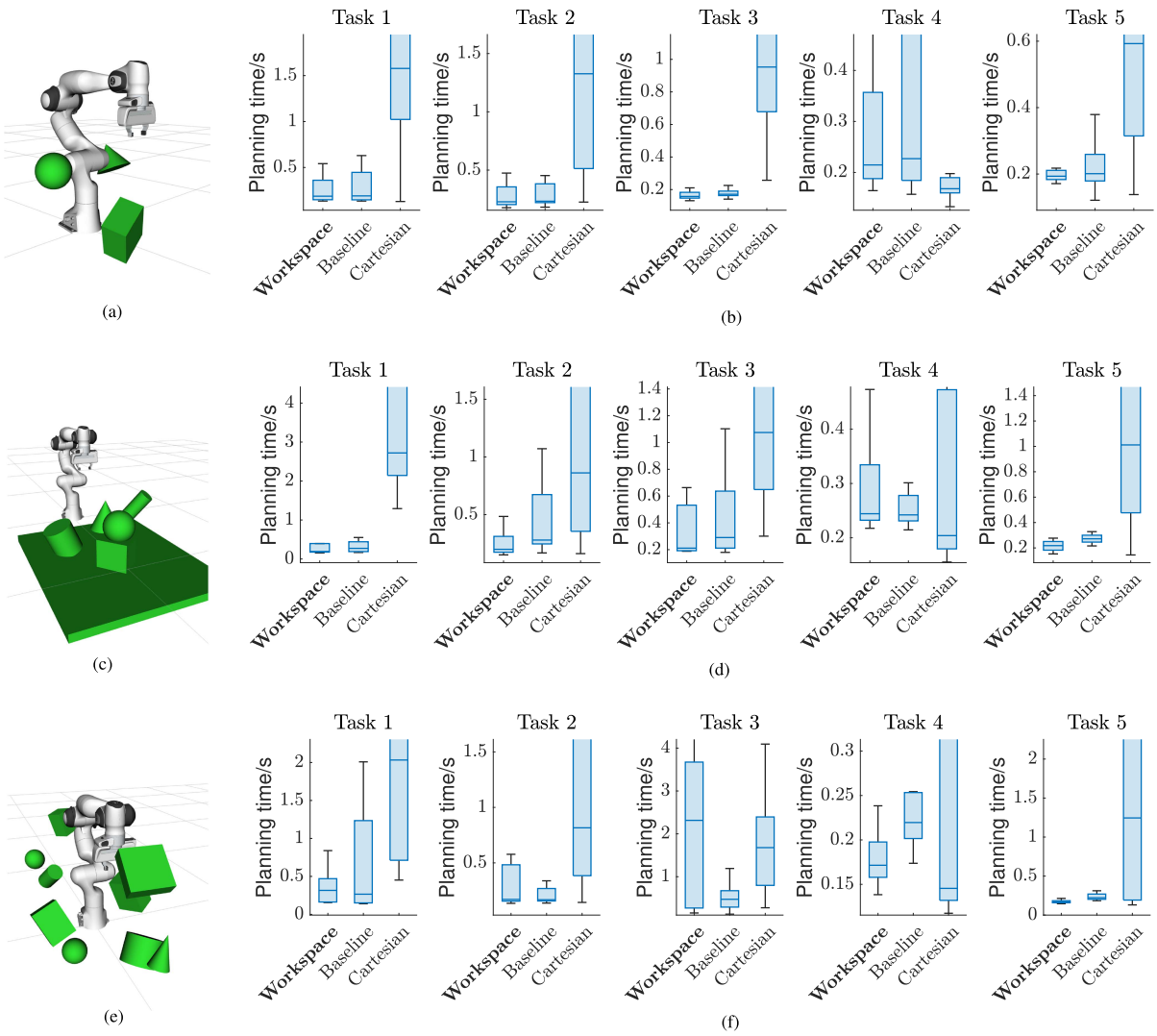


Fig. 12. Planning scenes and computational time comparisons for different STOMP variants. Each row shows the results in one planning scene, which is demonstrated in (a), (c), and (e), respectively. Figures (b), (d), and (f) show the planning time for the compared planners in different tasks as boxplots. Ours (Workspace-STOMP) is highlighted by bold letters. (a) Sparse scene. (b) Planning time comparisons in a sparse scene. (c) Cluttered scene. (d) Planning time comparisons in a cluttered scene. (e) Dense scene. (f) Planning time comparisons in dense scene.

include the trajectory similarity cost. When further considering the task success rate, ours performs much better in terms of the expected problem-solving time. In many cases, e.g., tasks 1 in all the scenes, tasks 3 and 5 in the dense scene, ours takes the lead with a large margin. Also, for task 3 in the dense scene, both compared baselines fail to complete the task after planning, but ours is able to solve the problem successfully in several trials.

VI. Physical Experiments

A robotic system is proposed, which includes PRIMP, Workspace-STOMP, and robot imagination [20]. The tasks and the corresponding motion primitives are the same in Table I. Physical experiments using the Franka Emika Panda robot are conducted.

A. Workflow

The general workflow of the proposed robotic system is shown in Fig. 13. For each motion primitive, human operators first conduct several demonstrations by dragging the robot end effector to fulfill the specific task. The trajectory of the end effector poses for each demonstration is recorded. For a new planning request, *key poses* for the robot are generated from manual inputs, ArUco tags [65], or robot imagination module (as in Section VI-D). A set of key pose candidates is then fed into PRIMP to condition the trajectory probabilistic distribution. The learned distribution is then used to guide the STOMP planner with new planning scenes, which include novel obstacles. Once a feasible trajectory is found by Workspace-STOMP, the robot executes the planned motion to fulfill the designated task. If there is no feasible trajectory, more key pose candidates are generated for replanning.

B. 3-D Reconstruction of the Planning Scene

A new scenario is obtained by 3-D reconstructions using an RGB+D camera. It is mounted at the robot end effector and moved to several predefined configurations (the details

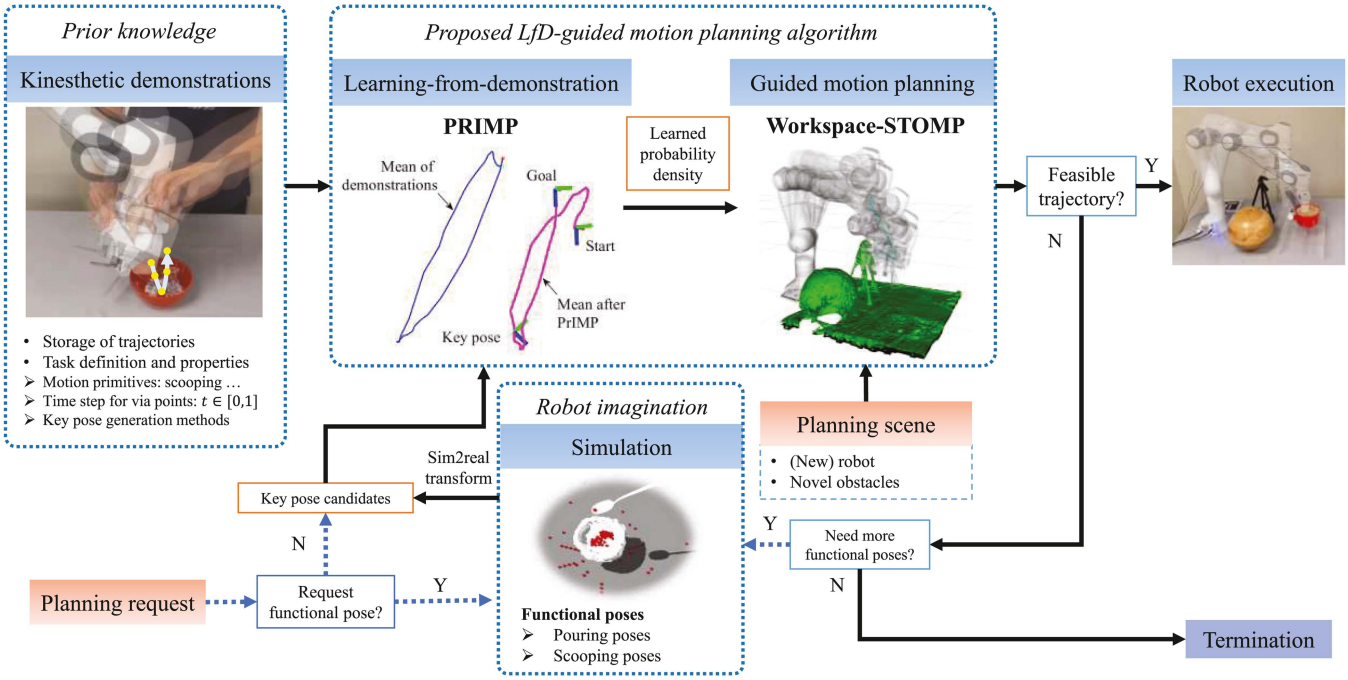


Fig. 13. Workflow of the proposed robotic system including PRIMP, Workspace-STOMP, and robot imagination.

are referred to the work in [20]). The object of interest, i.e., container or drawer, is segmented from the planning scene by placing it within a transparent table whose size is predefined and pose is indicated by an ArUco tag. After the scanning, the object of interest and obstacles outside the transparent table are constructed as mesh models separately. The object of interest is then loaded into the robot imagination module for tasks such as pouring and scooping. In the planning module, only the novel obstacles are loaded into the planning scene.

C. Prior Knowledge for Tasks and Tools

Prior knowledge of each task includes the method to obtain *functional poses*, *key poses*, and transformation from functional pose to key pose. The functional pose is obtained by 1) robot imagination, such as the pouring pose and scooping pose; or 2) the tag that indicates the start/goal pose for transporting or opening. The key pose is defined with respect to the specific link of the robot arm that is the same as the recorded one during the demonstrations. One key pose is the start pose, which is predefined for each task. Other key poses are generated depending on whether robot imagination is applied. For the tasks that require robot imagination, i.e., pouring and scooping, only the time parameter is defined a priori. But for tasks such as opening, the goal pose is computed as prior knowledge by considering the dimension of the moving part and the mode of movement. The prior knowledge for relative transformation from the functional pose to the key pose is defined for each tool, i.e., through the geometries of the handle, spoon, and cup and the corresponding grasping point for the robot gripper. Table XIII summarizes the prior knowledge for each task, including the process that generates key poses and the corresponding time parameter (i.e., $t \in [0, 1]$).

TABLE XIII
PRIOR KNOWLEDGE FOR DIFFERENT TASKS

Task ID	Key pose	$t \in [0, 1]$	Generation method
1	Start	0.0	Certain distance above object
	Pouring	1.0	Pouring imagination
2	Start	0.0	ArUco tag
	Goal	1.0	ArUco tag
3	Start	0.0	Certain distance above object
	Scooping	0.5	Scooping imagination
	Goal	1.0	Certain distance above object
4	Start	0.0	ArUco tag
	Goal	1.0	Computed from object geometry
5	Start	0.0	ArUco tag
	Goal	1.0	Computed from object geometry

D. Robot Imagination

One of the most essential steps to generalize skills in different situations is to properly define the via points. As is shown in Table XIII, the via points can be manually defined and retrieved from visual features for certain tasks such as transporting or door opening. However, for pouring and scooping, the motions are more complex, and the poses that afford successful executions are hard to determine. This refers to the study of object affordance learning. In our previous work, the object affordances are learned through physics-based simulation with predefined motions, which is named as *robot imagination*. The methods provide improvisational acquisition of the key points to successfully complete a certain task in unseen situations, with a high success

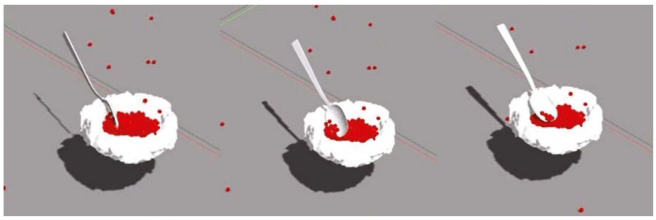


Fig. 14. Robot imagination process for scooping. The container is represented by the white meshed object.

rate [20], [66], [67]. Those key points are named as *functional poses* of the tool used to complete such a task. In this work, the via points for pouring and scooping tasks are determined by robot imagination trials, which are conducted using the Gazebo physics simulator. The pouring imagination applies the same principle as in [20] while the scooping imagination is novel in this work. The process for simulating the scooping tasks is described as follows.

The tool for scooping is a common spoon model, whose center is defined as the center of the rim of its head part. After the object of interest is loaded into the simulator, particles are dropped above the object's bounding box. Once there are particles contained inside, the one with the minimum height is recorded, denoted as the *bottom particle*. Its position is treated as the potential scooping position. Then, the spoon is placed on top of this bottom particle with random variations in the $x-y$ direction. The orientation is sampled as a uniformly distributed random rotation around the global z-axis. Before scooping, the spoon is rotated for 90° around its longer principle axis so that the head is facing the side. Then, the spoon is slowly rotated around the longer principle axis until the head is facing the top (as shown in Fig. 14). The current pose is a candidate for the *functional (scooping) pose*. Then, the spoon is lifted above the object to count the particles. Once there is any particle within the spoon head, the functional pose candidate is recorded.

The obtained functional poses are transformed into key poses for the robot by prior knowledge defined in Section VI-C and will be used in PRIMP. The number of attempts (N_{attempt}) and the maximum number of functional poses (N_{pose}) are defined by users. The imagination process is considered a success if $N_{\text{pose}} > 0$ after N_{attempt} trials, and failure otherwise.

E. Results

The objects and tools used for each task are shown in Fig. 15. For the demonstrations and robot executions, different objects and tools are used. The experimental results for the tasks in Table I are demonstrated in Fig. 16. One example is shown for each task.

VII. Discussion and Future Work

PRIMP learns the probability distribution and generalizes to novel situations in the full workspace of the robot manipulator. The learned trajectory distribution can be transferred to other robots with the conditioning by workspace density. Afterward, the trajectory can be shifted to the higher probability region in the robot workspace. When there is a rich set of demonstrations, the probability density gives more information for the whole motion where critical parts can be reflected. Even when there is only one demonstration, an initial covariance can be manually defined without affecting the generalization process for novel situations. Another feature of PRIMP is that the covariance is computed for the relative poses between two adjacent time steps. This is different from many existing probabilistic methods, which compute covariances for absolute states. It makes possible to adapt to the extrapolation case, where via points are out of the range of demonstrations. When there is a change of the start pose, the whole trajectory will be shifted accordingly, but the internal connections between poses and the trajectory shape will remain invariant. When other via points are far from the learned distribution, PRIMP can still perform well to both reach the target pose and maintain the trajectory shape. In terms of parameters, ProMP requires an explicit definition of the basis function and Orientation-KMP uses kernel tricks with parameters that affect performance significantly. However, the proposed PRIMP does not have any parameters to tune.

The proposed Workspace-STOMP uses the learned trajectory distribution as guidance to plan a feasible path. The covariance information provided by PRIMP gives more flexibility in varying the samples and guides the optimization through critical regions. For example, when the covariance is large, the trajectory can deviate more from the reference and focuses more on collision avoidance. In a certain region when the covariance is small, the planner will maintain the shape much closer to the reference. This cannot be fulfilled when only the mean trajectory is considered in the cost.

The robot imagination module aims at generating key poses for PRIMP, especially for pouring and scooping that involves more complex motions. When combining with the functionality of objects such as bowls, spoons, or cups, the definition of these via points is natural and explainable. By hard-coding and simulating the pouring and scooping processes, the critical functional pose is well-defined to assist the whole framework. Physical experiments demonstrate the feasibility of the imagined key poses, with the guidance of which the proposed PRIMP method and Workspace-STOMP planner are able to successfully transfer the human skills to the robot for the same task in different unseen scenarios.

The proposed methods still have limitations. For example, the imagination is still preprogrammed, which might not truly reflect human behaviors. The mean trajectory learned from PRIMP is sometimes not smooth enough, which requires STOMP to optimize. Also, when there is an anomaly in the demonstration set, PRIMP might have undesirable performance. To potentially resolve these limitations, future work includes fusing demonstrations into the robot imagination module; adding velocity and/or acceleration into the state vector; adding force information in the probabilistic model; combining with other optimization-based planning and control baselines, such as LQR, CHOMP, or TrajOpt, for guided planning; and investigating ways to detect and filter data outliers before encoding demonstrations.

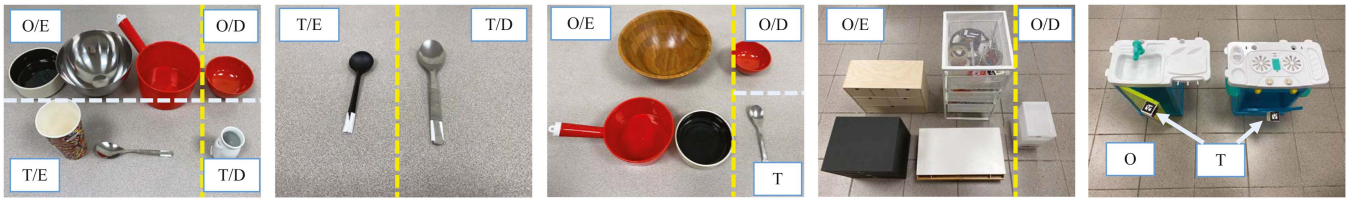


Fig. 15. Objects of interest and tools used for each task. O and T stand for "object of interest" and "tool," respectively; E and D stand for "robot execution" and "demonstration," respectively. The format X/Y means "X used in the Y process." And the ones without E or D are used in both processes.

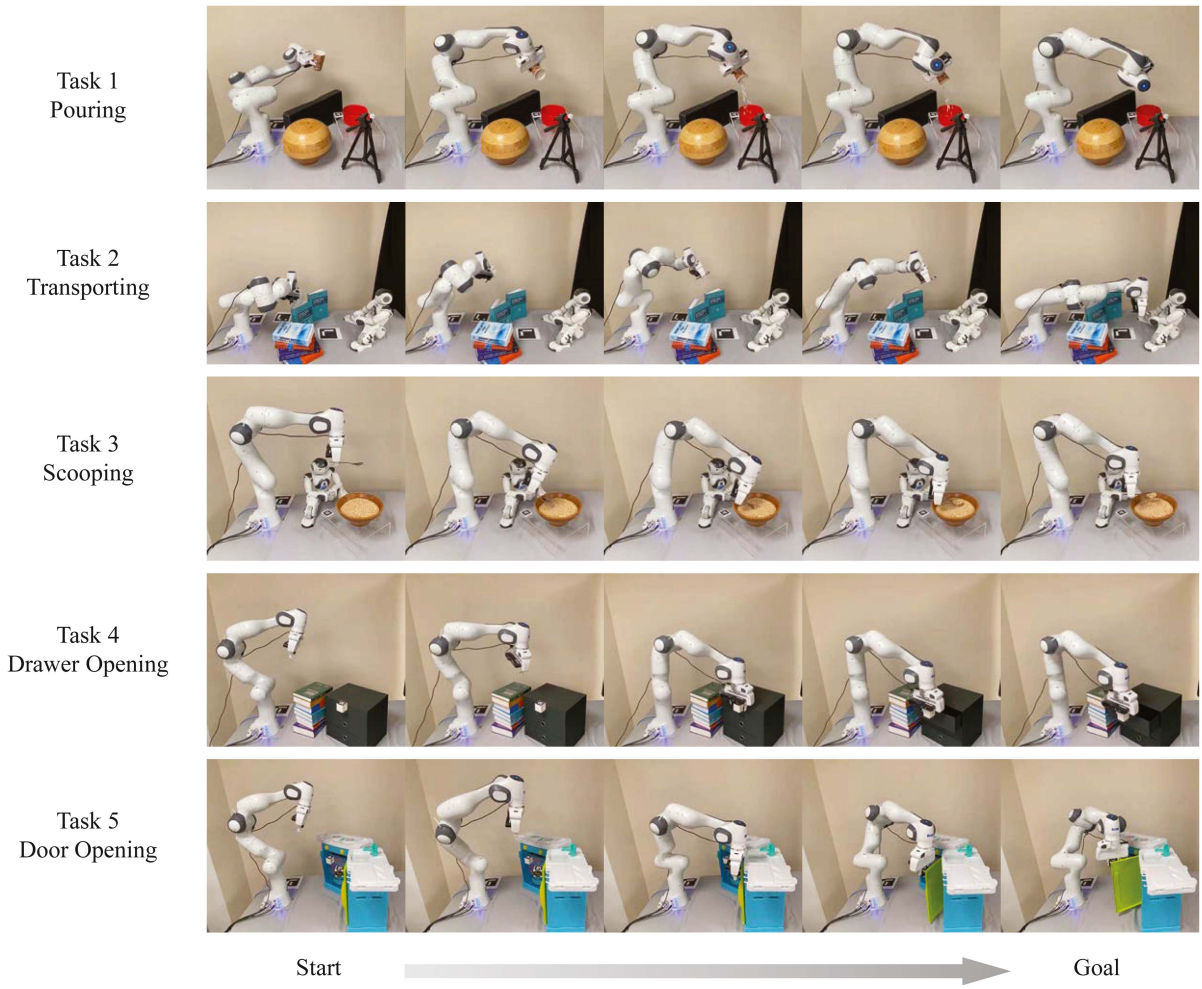


Fig. 16. Experiment results using the proposed robotic system including PRIMP, Workspace-STOMP, and robot imagination.

VIII. Conclusion

This article presents PRIMP, an LfD method that computes the probability distribution in the robot workspace with taught trajectories and simulation-informed actions. It only requires a few or even a single demonstration and is able to adapt to new via points (including start, goal, and any point in between), a change of viewing frame, and robot-specific workspace density. The learned trajectory distribution is then used to guide the STOMP motion planner to avoid novel obstacles, resulting in the *Workspace-STOMP* planner. Benchmark studies show the superiority among different popular LfD methods and guided motion planners. The applicability is demonstrated experimentally in a novel robotic system with the study of object affordance.

$$\Sigma_{1,\dots,n}^{-1} = \begin{pmatrix} \Sigma_{0,1}^{-1} + \widetilde{\Sigma}_{1,2} & -\mathrm{Ad}_{1,2}^{-T}\Sigma_{1,2}^{-1} & 0 & 0 & \dots & 0 \\ -\Sigma_{1,2}^{-1}\mathrm{Ad}_{1,2}^{-T} & \Sigma_{1,2}^{-1} + \widetilde{\Sigma}_{2,3} & -\mathrm{Ad}_{2,3}^{-T}\Sigma_{2,3}^{-1} & 0 & \dots & 0 \\ 0 & -\Sigma_{2,3}^{-1}\mathrm{Ad}_{2,3}^{-T} & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \dots & \dots & 0 \\ \dots & \dots & \dots & \dots & \Sigma_{n-2,n-1}^{-1} + \widetilde{\Sigma}_{n-1,n} & -\mathrm{Ad}_{n-1,n}^{-T}\Sigma_{n-1,n}^{-1} \\ 0 & \dots & \dots & 0 & -\Sigma_{n-1,n}^{-1}\mathrm{Ad}_{n-1,n}^{-1} & \Sigma_{n-1,n}^{-1} \end{pmatrix}. \tag{37}$$

APPENDIX

A. Derivations of Joint Probability That Encodes the Demonstrations

The absolute mean trajectory is denoted as $\{\mu_0, \mu_1, \dots, \mu_n\}$ and relative covariance is $\{\Sigma_{0,1}, \Sigma_{1,2}, \dots, \Sigma_{n-1,n}\}$. From (7), if all the poses are subject to Gaussian distribution with small deviations, explicit results can be shown as follows. First

$$\rho(g_{i+1}|g_i) \propto \exp\left(-\frac{1}{2}\left\|\log^{\vee}[(\mu_i^{-1}\mu_{i+1})^{-1}(g_i^{-1}g_{i+1})]\right\|_{\Sigma_{i,i+1}}^2\right). \tag{31}$$

Defining free variables that measure the deviation of the true poses to their mean as $g_i = \mu_i \circ \exp(\widehat{\boldsymbol{x}}_i)$. Then, the matrix logarithm term can be approximated as

$$\begin{aligned} &\log^{\vee}\left[(\mu_i^{-1}\mu_{i+1})^{-1}\left((\mu_i \exp(\widehat{\boldsymbol{x}}_i))^{-1}(\mu_{i+1}\exp(\widehat{\boldsymbol{x}}_{i+1}))\right)\right] \\ =\ &\log^{\vee}\left[(\mu_i^{-1}\mu_{i+1})^{-1}\exp^{-1}(\widehat{\boldsymbol{x}}_i)(\mu_i^{-1}\mu_{i+1})\exp(\widehat{\boldsymbol{x}}_{i+1})\right] \\ \approx\ &\boldsymbol{x}_{i+1} - \mathrm{Ad}_{i,i+1}\boldsymbol{x}_i. \end{aligned} \tag{32}$$

Then, substituting (32) into (31) and defining a joint variable $\boldsymbol{x}_{i,i+1} \doteq [\boldsymbol{x}_i^T, \boldsymbol{x}_{i+1}^T]^T$ gives

$$\rho(g_{i+1}|g_i) \propto \exp\left(-\frac{1}{2}\|\boldsymbol{x}_{i,i+1}\|_{\Sigma'_{i,i+1}}^2\right) \tag{33}$$

where

$$\Sigma'^{-1}_{i,i+1} = \begin{pmatrix} \mathrm{Ad}_{i,i+1}^{-1}\Sigma_{i,i+1}^{-1}\mathrm{Ad}_{i,i+1}^{-1} & -\mathrm{Ad}_{i,i+1}^{-T}\Sigma_{i,i+1}^{-1} \\ -\Sigma_{i,i+1}^{-1}\mathrm{Ad}_{i,i+1}^{-T} & \Sigma_{i,i+1}^{-1} \end{pmatrix} \in \mathbb{R}^{12\times 12}. \tag{34}$$

Finally, substituting (33) into (7) gives

$$\rho(g_1, g_2, \dots, g_n) \propto \exp\left(-\frac{1}{2}\sum_{i=1}^{n-1}\|\boldsymbol{x}_{i,i+1}\|_{\Sigma_{i,i+1}^{-1}}^2\right). \tag{35}$$

Stacking the variables into the full state vector gives

$$\rho(g_1, g_2, \dots, g_n) \propto \exp\left(-\frac{1}{2}\|\boldsymbol{x}_{1,\dots,n}\|_{\Sigma_{1,\dots,n}^{-1}}^2\right) \tag{36}$$

where

$$\boldsymbol{x}_{1,\dots,n} \doteq [\boldsymbol{x}_1^T, \dots, \boldsymbol{x}_i^T, \dots, \boldsymbol{x}_n^T]^T$$
$$\widetilde{\Sigma}_{i,i+1} \doteq \mathrm{Ad}_{i,i+1}\Sigma_{i,i+1}\mathrm{Ad}_{i,i+1}^T$$

and (37) shown at the top of this page

B. Derivations of Equivariant Property of the Joint Distribution of the Whole Trajectory

The conditional probability $\rho(g_{i+1}|g_i)$ in (31) under the change of view is derived as follows. When $h \in \mathrm{SE}(3)$ is the transformation from frame O to A, for any $g \in \mathrm{SE}(3)$ that is viewed in O, $g^o = h^{-1}gh$ is the conjugated element that is viewed in frame A. Then, by direct computations, the matrix logarithm within the squared weighted norm is derived as

$$\begin{aligned} &\log^{\vee}\left((\mu_i^{-1}\mu_{i+1})^{-1}(g_i^{-1}g_{i+1})\right) \\ =\ &\log^{\vee}\left(h(\mu_i^{o-1}\mu_{i+1}^o)^{-1}g_i^{o-1}g_{i+1}^o h^{-1}\right) \\ =\ &\log^{\vee}\left(h(\mu_i^{o-1}\mu_{i+1}^o)^{-1}\exp(-\boldsymbol{x}_i^o)(\mu_i^{o-1}\mu_{i+1}^o)\exp(\boldsymbol{x}_{i+1}^o)h^{-1}\right) \\ =\ &\mathrm{Ad}_h\log^{\vee}\left((\mu_i^{o-1}\mu_{i+1}^o)^{-1}\exp(-\boldsymbol{x}_i^o)(\mu_i^{o-1}\mu_{i+1}^o)\exp(\boldsymbol{x}_{i+1}^o)\right). \end{aligned} \tag{38}$$

With the assumption of small deviations, i.e., $\|\boldsymbol{x}_i^o\|_2 \ll 1$, (31) can be approximated as

$$\rho(g_{i+1}|g_i) \propto \exp\left(-\frac{1}{2}\left\|\mathrm{Ad}_h\left(\boldsymbol{x}_{i+1}^o - \mathrm{Ad}_{i,i+1}^o\boldsymbol{x}_i^o\right)\right\|_{\Sigma_{i,i+1}^{-1}}^2\right) \tag{39}$$

which is equivalent to

$$\rho(g_{i+1}|g_i) \propto \exp\left(-\frac{1}{2}\left\|\boldsymbol{x}_{i+1}^o - \mathrm{Ad}_{i,i+1}^o\boldsymbol{x}_i^o\right\|_{\Sigma_{i,i+1}^{o-1}}^2\right) \tag{40}$$

where

$$\Sigma_{i,i+1}^o \doteq \mathrm{Ad}_h^{-1}\Sigma_{i,i+1}\mathrm{Ad}_h^{-T}.$$

ACKNOWLEDGMENT

The authors would like to thank Dr. Seng-Beng Ho and Mr. Jikai Ye for their useful discussions. The ideas expressed in this article are solely those of the authors.

REFERENCES

[1] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Ann. Rev. Control, Robot., Auton. Syst.*, vol. 3, pp. 297–330, 2020.

[2] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Springer Handbook Robot.* Berlin, Germany: Springer, 2008, pp. 1371–1394.

[3] W. Jin, T. D. Murphey, D. Kulić, N. Ezer, and S. Mou, "Learning from sparse demonstrations," *IEEE Trans. Robot.*, vol. 39, no. 1, pp. 645–644, Feb. 2023.

[4] J. Zhu, M. Gienger, and J. Kober, "Learning task-parameterized skills from few demonstrations," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 4063–4070, Apr. 2022.

[5] M. J. Zeestraten, I. Havoutis, J. Silvério, S. Calinon, and D. G. Caldwell, "An approach for imitation learning on Riemannian manifolds," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1240–1247, Jul. 2017.

[6] S. Calinon, "Gaussians on Riemannian manifolds: Applications for robot learning and adaptive control," *IEEE Robot. Autom. Mag.*, vol. 27, no. 2, pp. 33–45, Jun. 2020.

[7] Y. Huang, F. J. Abu-Dakka, J. Silvério, and D. G. Caldwell, "Toward orientation learning and adaptation in Cartesian space," *IEEE Trans. Robot.*, vol. 37, no. 1, pp. 82–98, Feb. 2021.

[8] B. Ti, A. Razmjoo, Y. Gao, J. Zhao, and S. Calinon, "A geometric optimal control approach for imitation and generalization of manipulation skills," *Robot. Auton. Syst.*, vol. 164, 2023, Art. no. 104413.

[9] S. Auddy, J. Hollenstein, M. Saveriano, A. Rodríguez-Sánchez, and J. Piater, "Continual learning from demonstration of robotics skills," *Robot. Auton. Syst.*, vol. 165, 2023, Art. no. 104427.

[10] G. S. Chirikjian and A. B. Kyatkin, *Harmonic Analysis for Engineers and Applied Scientists: Updated and Expanded Edition*. New York, NY, USA: Dover, 2016.

[11] G. S. Chirikjian, "Modeling loop entropy," *Methods Enzymol.*, vol. 487, pp. 99–132, 2011.

[12] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Robot placement based on reachability inversion," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 1970–1975.

[13] S. Jauhri, J. Peters, and G. Chalvatzaki, "Robot learning of mobile manipulation with reachability behavior priors," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 8399–8406, Jul. 2022.

[14] D. Koert, G. Maeda, R. Lioutikov, G. Neumann, and J. Peters, "Demonstration based trajectory optimization for generalizable robot motions," in *Proc. IEEE-RAS 16th Int. Conf. Humanoid Robots*, 2016, pp. 515–522.

[15] G. Ye and R. Alterovitz, "Guided motion planning," in *Proc. Robot. Res.*, 2017, pp. 291–307.

[16] B. Magyar, N. Tsiogkas, B. Brito, M. Patel, D. Lane, and S. Wang, "Guided stochastic optimization for motion planning," *Front. Robot. AI*, vol. 6, 2019, Art. no. 105.

[17] M. Dobiš, M. Dekan, A. Sojka, P. Beňo, and F. Duchoň, "Cartesian constrained stochastic trajectory optimization for motion planning," *Appl. Sci.*, vol. 11, no. 24, 2021, Art. no. 11712.

[18] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 4569–4574.

[19] S.-B. Ho, "A general framework for the representation of function and affordance: A cognitive, causal, and grounded approach, and a step toward AGI," 2022, *arXiv:2206.05273*.

[20] H. Wu and G. S. Chirikjian, "Can I pour into it? Robot imagining open containability affordance of previously unseen objects via physical simulations," *IEEE Robot. Autom. Lett.*, vol. 6, no. 1, pp. 271–278, Jan. 2021.

[21] V. Vosylius and E. Johns, "Where to start? Transferring simple skills to complex environments," in *Proc. 6th Annu. Conf. Robot Learn.*, vol. 205, 2022, pp. 471–481.

[22] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Comput.*, vol. 25, no. 2, pp. 328–373, 2013.

[23] Y. Fanger, J. Umlauft, and S. Hirche, "Gaussian processes for dynamic movement primitives with application in knowledge-based cooperation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 3913–3919.

[24] Z. Li, T. Zhao, F. Chen, Y. Hu, C.-Y. Su, and T. Fukuda, "Reinforcement learning of manipulation and grasping using dynamical movement primitives for a humanoid-like mobile manipulator," *IEEE/ASME Trans. Mechatron.*, vol. 23, no. 1, pp. 121–131, Feb. 2018.

[25] A. Duan, R. Camoriano, D. Ferigo, D. Calandriello, L. Rosasco, and D. Pucci, "Constrained DMPs for feasible skill learning on humanoid robots," in *Proc. IEEE-RAS 18th Int. Conf. Humanoid Robots*, 2018, pp. 1–6.

[26] M. Saveriano, F. J. Abu-Dakka, A. Kramberger, and L. Peternel, "Dynamic movement primitives in robotics: A tutorial survey," *Int. J. Robot. Res.*, vol. 42, pp. 1133–1184, 2021.

[27] J. Peters and S. Schaal, "Reinforcement learning of motor skills with policy gradients," *Neural Netw.*, vol. 21, no. 4, pp. 682–697, 2008.

[28] J. Kober and J. Peters, "Policy search for motor primitives in robotics," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 21, 2008 .

[29] S. Calinon, D. Bruno, and D. G. Caldwell, "A task-parameterized probabilistic model with minimal intervention control," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 3339–3344.

[30] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," *Intell. Service Robot.*, vol. 9, pp. 1–29, 2016.

[31] S. Calinon, Z. Li, T. Alizadeh, N. G. Tsagarakis, and D. G. Caldwell, "Statistical dynamical systems for skills acquisition in humanoids," in *Proc. IEEE-RAS 12th Int. Conf. Humanoid Robots*, 2012, pp. 323–329.

[32] F. Meier and S. Schaal, "A probabilistic representation for dynamic movement primitives," 2016, *arXiv:1612.05932*.

[33] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, "Using probabilistic movement primitives in robotics," *Auton. Robots*, vol. 42, no. 3, pp. 529–551, 2018.

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

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

[36] T. Kulak, H. Girgin, J.-M. Odobez, and S. Calinon, "Active learning of Bayesian probabilistic movement primitives," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 2163–2170, Apr. 2021.

[37] Y. Zhou, J. Gao, and T. Asfour, "Learning via-point movement primitives with inter-and extrapolation capabilities," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 4301–4308.

[38] Y. Huang, L. Rozo, J. Silvério, and D. G. Caldwell, "Kernelized movement primitives," *Int. J. Robot. Res.*, vol. 38, no. 7, pp. 833–852, 2019.

[39] A. Duan, I. Batzianoulis, R. Camoriano, L. Rosasco, D. Pucci, and A. Billard, "A structured prediction approach for robot imitation learning," *Int. J. Robot. Res.*, vol. 43, no. 2, pp. 113–133, 2024.

[40] A. Ude, B. Nemec, T. Petrić, and J. Morimoto, "Orientation in Cartesian space dynamic movement primitives," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 2997–3004.

[41] F. J. Abu-Dakka, M. Saveriano, and V. Kyrki, "A unified formulation of geometry-aware dynamic movement primitives," 2022, *arXiv:2203.03374*.

[42] J. Silvério, L. Rozo, S. Calinon, and D. G. Caldwell, "Learning bimanual end-effector poses from demonstrations using task-parameterized dynamical systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 464–470.

[43] L. Rozo and V. Dave, "Orientation probabilistic movement primitives on Riemannian manifolds," in *Proc. Conf. Robot Learn.*, 2022, pp. 373–383.

[44] M. Rana, M. Mukadam, S. R. Ahmadzadeh, S. Chernova, and B. Boots, "Towards robust skill generalization: Unifying learning from demonstration and motion planning," in *Proc. 1st Annu. Conf. Robot Learn.*, vol. 78 2017, pp. 109–188.

[45] M. A. Rana, M. Mukadam, S. R. Ahmadzadeh, S. Chernova, and B. Boots, "Learning generalizable robot skills from demonstrations in cluttered environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4655–4660.

[46] M. Zucker et al., "CHOMP: Covariant Hamiltonian optimization for motion planning," *Int. J. Robot. Res.*, vol. 32, no. 9/10, pp. 1164–1193, 2013.

[47] T. Osa, A. M. G. Esfahani, R. Stolkin, R. Lioutikov, J. Peters, and G. Neumann, "Guiding trajectory optimization by demonstrated distributions," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 819–826, Apr. 2017.

[48] R. A. Shyam, P. Lightbody, G. Das, P. Liu, S. Gomez-Gonzalez, and G. Neumann, "Improving local trajectory optimisation using probabilistic movement primitives," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 2666–2671.

[49] C. Quintero-Pena, C. Chamzas, Z. Sun, V. Unhelkar, and L. E. Kavraki, "Human-guided motion planning in partially observable environments," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 7226–7232.

[50] G. S. Chirikjian, R. Mahony, S. Ruan, and J. Trumpf, "Pose changes from a different point of view," *J. Mechanisms Robot.*, vol. 10, no. 2, 2018.

[51] T. W. Mitchel, S. Ruan, and G. S. Chirikjian, "Signal alignment for humanoid skeletons via the globally optimal reparameterization algorithm," in *Proc. IEEE-RAS 18th Int. Conf. Humanoid Robots*, 2018, pp. 217–223.

[52] A. Vakanski, I. Mantegh, A. Irish, and F. Janabi-Sharifi, "Trajectory learning for robot programming by demonstration using hidden Markov model and dynamic time warping," *IEEE Trans. Syst., Man, Cybern., Part B.*, vol. 42, no. 4, pp. 1039–1052, Aug. 2012.

[53] G. S. Chirikjian, "Bootstrapping globally optimal variational calculus solutions," *Calculus Variations Partial Differ. Equ.*, vol. 62, no. 2, pp. 1–33, 2023.

[54] M. K. Ackerman and G. S. Chirikjian, "A probabilistic solution to the AX=XB problem: Sensor calibration without correspondence," in *Proc. Int. Conf. Geometric Sci. Inf.*, 2013, pp. 693–701.

[55] G. S. Chirikjian, *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*, vol. 2, Berlin, Germany: Springer, 2011.
[56] Y. Wang and G. S. Chirikjian, "Nonparametric second-order theory of error propagation on motion groups," *Int. J. Robot. Res.*, vol. 27, no. 11/12, pp. 1258–1273, 2008.
[57] I. Ebert-Uphoff and G. S. Chirikjian, "Inverse kinematics of discretely actuated hyper-redundant manipulators using workspace densities," in *Proc. IEEE Int. Conf. Robot. Automat.*, vol. 1,1996, pp. 139–145.
[58] G. S. Chirikjian and I. Ebert-Uphoff, "Numerical convolution on the Euclidean group with applications to workspace generation," *IEEE Trans. Robot. Autom.*, vol. 14, no. 1, pp. 123–136, Feb. 1998.
[59] Y. Wang and G. S. Chirikjian, "Workspace generation of hyper-redundant manipulators as a diffusion process on SE(N)," *IEEE Trans. Robot. Autom.*, vol. 20, no. 3, pp. 399–408, Jun. 2004.
[60] P. Smith, T. Drummond, and K. Roussopoulos, "Computing MAP trajectories by representing, propagating and combining PDFs over groups," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2003, pp. 1275–1282.
[61] T. D. Barfoot and P. T. Furgale, "Associating uncertainty with three-dimensional poses for use in estimation problems," *IEEE Trans. Robot.*, vol. 30, no. 3, pp. 679–693, Jun. 2014.
[62] S. Chitta, I. Sucan, and S. Cousins, "Moveit! [ROS topics]," *IEEE Robot. Autom. Mag.*, vol. 19, no. 1, pp. 18–19, Mar. 2012.
[63] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with Gaussian mixture models," *IEEE Trans. Robot.*, vol. 27, no. 5, pp. 943–957, Oct. 2011.
[64] J.-M. Lien, "Hybrid motion planning using Minkowski sums," in *Proc. Robot.: Sci. Syst. IV*, 2008, pp. 97–104.
[65] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognit.*, vol. 47, no. 6, pp. 2280–2292, 2014.
[66] H. Wu, X. Meng, S. Ruan, and G. S. Chirikjian, "Put the bear on the chair! Intelligent robot interaction with previously unseen chairs via robot imagination," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2022, pp. 6276–6282.
[67] X. Meng, H. Wu, S. Ruan, and G. S. Chirikjian, "Prepare the chair for the bear! Robot imagination of sitting affordance to reorient previously unseen chairs," *IEEE Robot. Autom. Lett.*, vol. 8, no. 10, pp. 6515–6522, Oct. 2023.

Sipu Ruan (Member, IEEE) received the bachelor's degree in mechanical engineering from the Harbin Institute of Technology, Harbin, China, in 2015, and the master's degree in robotics and the Ph.D. degree in mechanical engineering from Johns Hopkins University, Baltimore, MD, USA, in 2017 and 2020, respectively.

From 2021 to 2024, he was a Research Fellow with the Department of Mechanical Engineering, National University of Singapore, Singapore. His research interests include robot motion and path planning, imitation learning, computational kinematics, and service robots.

Weixiao Liu (Member, IEEE) received the bachelor's degree in mechanical engineering and the master's degree in mechatronics from the Dalian University of Technology, Dalian, China, in 2015 and 2018, respectively, and the master's and Ph.D. degrees in mechanical engineering (robotics) from Johns Hopkins University, Baltimore, MD, USA, in 2020 and 2023, respectively.

During 2021–2023, he was a Research Engineer with the National University of Singapore. In 2023, he joined Tencent America, Los Angeles, CA, USA, where he is currently a Senior Researcher. His research interests include probabilistic modeling, Bayesian inference, computational geometry, and computer vision, currently focusing on geometric shape abstraction and generative AI.

Xiaoli Wang (Graduate Student Member, IEEE) received the bachelor's degree in biomedical engineering from the Beijing Institute of Technology, Beijing, China, in 2020, and the M.Eng. degree in biomedical engineering in 2021 from the National University of Singapore, Singapore, where she is currently working toward the Ph.D. degree in Mechanical Engineering.

Her research interests include probabilistic collision detection, planning under uncertainty, object affordance reasoning, and robotic manipulation.

Xin Meng (Graduate Student Member, IEEE) received the B.Eng. degree and the M.Res. degree in mechanical engineering from Beihang University, Beijing, China, in 2017 and 2020, respectively. She is currently working toward the Ph.D. degree in Mechanical Engineering, National University of Singapore, Singapore.

Her research interests include robot imagination, object affordance reasoning, and robot manipulation.

Gregory S. Chirikjian (Fellow, IEEE) received the BSE degree in Engineering Mechanics and MSE in Mechanical Engineering from Johns Hopkins University, Baltimore, MD, USA, in 1988, and the PhD degree in Applied Mechanics from the California Institute of Technology, Pasadena, CA, USA, in 1992.

From 1992 to 2021, he was with the Faculty of the Department of Mechanical Engineering, Johns Hopkins University, attaining the rank of Full Professor in 2001. Additionally, from 2004 to 2007, he was a Department Chair. In 2019, he joined the National University of Singapore, where he was the Head of the Mechanical Engineering Department until summer of 2023. Since 2024, he has been the Chair of the Mechanical Engineering Department, University of Delaware, Newark, DE, USA. He is the author of more than 250 journals and conference papers and the primary author of three books, including *Engineering Applications of Noncommutative Harmonic Analysis* (2001) and *Stochastic Models, Information Theory, and Lie Groups, Vols. 1+2* (2009, 2011). In 2016, an expanded edition of his 2001 book was published as a Dover book under a new title, *Harmonic Analysis for Engineers and Applied Scientists*. His research interests include robotics, applications of group theory in state estimation, information-theoretic inequalities, and applied mathematics more broadly.

Dr. Chirikjian is a 1993 National Science Foundation Young Investigator and a 1994 Presidential Faculty Fellow. During 2014–2015, he was a Program Director for the US National Robotics Initiative, which included responsibilities in the robust intelligence cluster in the Information and Intelligent Systems Division, CISE, NSF.