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

Sipu Ruan, Weixiao Liu, Xiaoli Wang, Xin Meng and Gregory S. Chirikjian*

***Abstract*—This paper proposes a novel framework for learning skills from one or a few demonstrations and generalizing skills to unseen objects, environments, and robots. The proposed learning-from-demonstration method, named "PRobabilistically-Informed Motion Primitives (PRIMP)", captures motion features by learning the probability distribution of the end effector trajectories in the 6D workspace that includes both positions and orientations. The learned trajectory distribution can adapt to new situations, such as novel via points and changes in the 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. The learned trajectory distribution is then used to guide an optimization-based motion planning algorithm to further help the robot avoid novel obstacles that are unseen during demonstrations while keeping the learned motion features. The proposed method is evaluated by several sets of benchmark experiments. PRIMP runs more than 5 times faster than existing state-of-the-art methods while generalizing trajectories more than twice as close to both the demonstrations and novel via points. It is then combined with our lab's robot imagination method that learns object affordances, illustrating the applicability of PRIMP to learn tool use through physical experiments.**

I. Introduction

For a robot to be truly intelligent to function in the real world, it needs the ability to learn from prior knowledge while adapting to unseen scenarios. The prior knowledge can be some simple pre-programmed 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 pre-program but are ubiquitous in household environments, like scooping powder (as in Figure 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 (PbD)* [2].

Many works on LfD encode trajectory models in the Euclidean space. Analytic methods like Dynamical movement primitives (DMP) [3] or reinforcement learning-based methods [4] [5] are able to encode the position, velocity, and acceleration of demonstrated motions, suitable for high-speed motions like hitting a ball. Another class of LfD method uses probabilistic models to handle the inherent uncertainty and variability in demonstrations, such as the probabilistic movement primitives (ProMP) [6] and the kernelized movement primitives (KMP) [7]. Recently, probabilistic methods have been extended to the probability distribution within a manifold to include the motion features of orientation, such as the Orientation-KMP [8], which includes orientation by quaternions. The learned probability distribution of the motion primitive can be generalized to novel via points and new scenarios. However, the performance of existing probabilistic methods relies on the choice of basis functions or kernels.

Robotic manipulation tasks, like scooping cereals, have complex and implicit constraints for the motion of the robot end-effector in the workspace. Consequently, 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 the concept of loop entropy [9], [10] and more recent work on inverse reachability mapping [11], [12]. It is further extended here into LfD with via points 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 usually conducted in a scene with only the robot and the objects with which it interacts. When adapting to a novel scene with unseen obstacles, only using LfD methods cannot guarantee a collision-free trajectory. Guided motion planning, which combines LfD and motion planning, has become popular in the recent decade [13]–[16]. The goal is to make the motion collision-free while keeping the critical features from the learned trajectory as many as possible. In this work, an optimization-based planner, *Stochastic Trajectory Optimization for Motion Planning (STOMP)* [17], is applied as the base framework. The reference trajectory is the mean of the learned trajectory distribution via PRIMP. 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*.

Apart from novel obstacles, the object that the robot interacts with might also be unseen or even from different categories. For example, the pouring task is demonstrated by

Sipu Ruan, Xiaoli Wang, Xin Meng and Gregory S. Chirikjian (mpegre@nus.edu.sg) are with Department of Mechanical Engineering, National University of Singapore, Singapore.

Gregory S. Chirikjian is currently at the University of Delaware.

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

* Address all correspondence to this author.

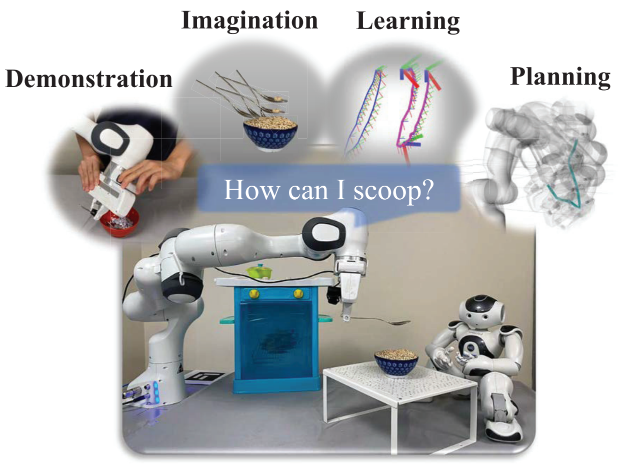


Fig. 1: Illustration for the general idea of this work. The robot arm is asked to use a spoon to scoop from an unseen bowl in a new environment. With the help of human demonstrations, imagination of object affordance, learning skills from the demonstrations, and motion planning, the robot is able to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

pouring powders from a cup to a bowl. Here, the cup is treated as a tool for pouring, and the bowl has the affordance of containing powders. 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, 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 [18]. Recent works proposed methods to learn object affordance, like pouring, via physics-based simulation [19], [20]. Our work applies this idea to learn the key pose of a task, *i.e.*, the pouring pose, which is used as a new goal or key pose for re-production. 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 through physical experiments.

The contributions of our work are listed as follows.

- A learning-from-demonstration method, PRIMP, is proposed to generate reference workspace trajectory distribution for basic motion primitives;
- 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.
- A novel robotic system that combines LfD, motion planning, and affordance-based simulation is proposed and physically demonstrated in a robot manipulator platform.

II. Probabilistically-Informed Motion Primitives

This section introduces the proposed LfD method, namely *PRobabilistically-Informed Motion Primitives (PRIMP)*. This is a probabilistic method to encode the trajectory that involves both the 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 in the 6D workspace is modeled as an element in Lie group G. Therefore, the full state is considered in a product space $G \times ... \times G$, resulting in a state vector of dimension $6N_{\text{step}}$. Consider a set of demonstrated trajectories $\left\{g_0^{(k)}, g_1^{(k)}, ..., .g_n^{(k)}\right\}$, where $g_i^{(k)} \in \text{SE}(3)$ is the i^{th} step in the k^{th} demonstration. 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.

A. General framework of PRIMP

Firstly, all the demonstrated trajectories are temporally aligned into the same time scale. After alignment, the probability distribution of the $(i+1)^{\text{th}}$ pose with respect to the i^{th} pose is approximated using a Lie-theoretic method (Sec. II-C). The computed initial mean and covariance is then encoded into the joint distribution of the whole trajectory (Sec. II-D). Several types of adaptations to novel scenarios are then introduced (Sec. II-E):

- when an intermediate pose is different from the given mean and has some uncertainties, a posterior distribution is computed to adapt to this new situation (Sec. II-E1);
- when there is a change of viewing frame, the encoded distribution can be adapted in the sense of equivariance (Sec. II-E2);
- 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 (Sec. II-E3).

B. Temporal Alignment for Demonstrations using Globally Optimal Reparameterization Algorithm (GORA)

Different demonstrations might have different speeds of execution, even when the trajectories have the same shape [21]. 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 essential. In our work, a variational calculus technique is applied to align multiple trajectories with global optimality in the temporal axis, which is named as Globally Optimal Reparameterization Algorithm (GORA) [22].

Suppose an SE(3) sequence $g(\tau)$ is parameterized by $\tau \in [0, 1]$. Here $\tau(t) : [0, 1] \to [0, 1]$ is a one-dimensional 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.*,

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

Algorithm 1: Globally Optimal Reparameterization Algorithm (GORA) for SE(3) sequence [22]

```
Inputs    : $g(\tau)$: Input SE(3) sequence
Parameter: $N_{\text{step}}$: Number of time steps
Outputs   : $g(\tau^*)$: Optimal reparameterized sequence
            $\tau^*(t)$: Optimal temporal parameter
1 Initialize $\tau$ and $t$ as uniform sampling within $[0, 1]$;
2 Compute $\mathfrak{g}(\tau)$ from Eq. (2);
3 for time step $i$ do
4     $F(\tau_i)$ = 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($\tau^*$, $g(\tau)$);
```

where $\dot{\tau} = d\tau/dt$. 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 $\mathcal{A} \in \mathbb{R}^{4\times4}$, $\|\mathcal{A}\|_W = \sqrt{\text{tr}\left(\mathcal{A}^T W \mathcal{A}\right)}$, for some symmetric matrix

$$W \doteq \begin{pmatrix} \frac{1}{2}\text{tr}(I)\mathbb{I}_3 - I & \mathbf{0} \\ \mathbf{0}^T & 1 \end{pmatrix} \in \mathbb{R}^{4\times4} ,$$

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.

With the integrated form as Eq. 2 and the boundary conditions $\tau(0) = 0$ and $\tau(1) = 1$, Eq. 1 has global minimization [22] [23]. The minimizer is $\tau^*(t) = F^{-1}(t)$, where $F^{-1}(\cdot)$ is the inverse function of

$$F(\tau^*) = \frac{\int_0^{\tau^*} \mathfrak{g}^{\frac{1}{2}}(\sigma)\, d\sigma}{\int_0^1 \mathfrak{g}^{\frac{1}{2}}(\sigma)\, d\sigma} = t\, . \tag{3}$$

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 Eq. (2). Both Lines 4 and 6 are the core computations of the algorithm, *i.e.*, Eq. (3). Here, the trapezoidal rule is applied for numerical integration 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 $\left\{g_i^{(k)}\right\}$ in SE(3) at each step i, the sample mean $\mu_i \in$ SE(3), is given by $\sum_{k=1}^m \log(\mu_i^{-1} g_i^{(k)}) = \mathbb{O}$, which can be iteratively solved [24]. Here, $\log(\cdot)$ 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 $(i+1)^{\text{th}}$ step given the i^{th} step. It is estimated by the set of relative poses, *i.e.*, $\left\{\Delta_{i,i+1}^{(k)} = \left(g_i^{(k)}\right)^{-1} g_{i+1}^{(k)}\right\}$. With this set, the sample covariance can be computed as $\Sigma_{i,i+1} = \frac{1}{m}\sum_{k=1}^m \log^\vee\left(\mu_{i,i+1}^{-1}\Delta_{i,i+1}^{(k)}\right) \log^{\vee T}\left(\mu_{i,i+1}^{-1}\Delta_{i,i+1}^{(k)}\right)$, where $\mu_{i,i+1}$ can be computed similar to μ_i but replace $g_i^{(k)}$ with the relative poses $\Delta_{i,i+1}^{(k)}$. The $\vee$ operator extracts the Lie algebra coefficients into a vector (as defined in [25]).

D. Probabilistic encoding of joint distribution on SE(3) trajectories

After computing the trajectory distribution with mean $\{\mu_0, \mu_1, ..., \mu_n\}$, $\mu_i \in$ SE(3) and covariance between adjacent steps $\{\Sigma_{0,1}, \Sigma_{1,2}, ..., \Sigma_{n-1,n}\}$, $\Sigma_{i,i+1} \in \mathbb{R}^{6\times6}$ from Sec. II-C, the joint distributions of the whole trajectory can be computed. Assuming the variation of i^{th} pose only depends on its two neighboring poses and $g_0 = \mu_0$ is fixed, the joint probability density can be expressed as

$$\rho(g_1, g_2, ..., g_n) = \prod_{i=0}^{n-1} \rho(g_{i+1}|g_i)\, , \tag{4}$$

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

Because the $\rho(g_i)$ is subjected to the Gaussian distribution with small variation, $\rho(g_1, g_2, ..., g_n)$ is also Gaussian distributed with joint variable $\boldsymbol{x}_{1,...,n}^T$, where $\boldsymbol{x}_i = \log^\vee(\mu_i^{-1} g_i)$, and joint covariance $\Sigma_{1,...,n} \in \mathbb{R}^{6n\times6n}$, where the $(i,i)^{\text{th}}$ block is expressed as $\Sigma_{1,...,n}(i,i)$. The non-zero blocks of $\Sigma_{1,...,n}^{-1}$ are

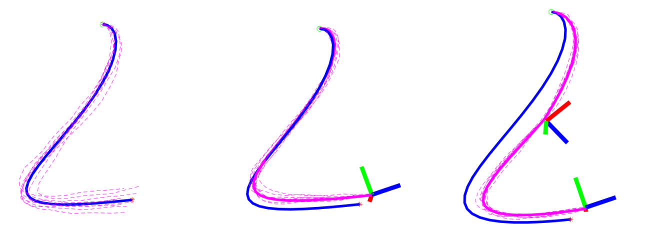
$$\begin{aligned} \Sigma_{1,...,n}^{-1}(i,i) &= \begin{cases} \Sigma_{i-1,i}^{-1} + \widetilde{\Sigma}_{i,i+1}^{-1} & (i \neq n) \\ \Sigma_{i-1,i}^{-1} & (i = n) \end{cases} \\ \Sigma_{1,...,n}^{-1}(i,i+1) &= -Ad_{i,i+1}^{-T}\Sigma_{i,i+1}^{-1} \;\; (i \neq n) \\ \Sigma_{1,...,n}^{-1}(i+1,i) &= -\Sigma_{i,i+1}^{-1} Ad_{i,i+1}^{-1} \;\; (i \neq n)\, . \end{aligned} \tag{5}$$

$Ad(g)$ is the adjoint matrix for the element g in a Lie group, defined as $Ad(g)\boldsymbol{x} = (g\hat{\boldsymbol{x}}g^{-1})^\vee$, where $\hat{\cdot}$ is the inverse operation of $\vee$ that maps a vector space into the Lie algebra. $Ad_{i,i+1} \doteq Ad(\mu_i^{-1}\mu_{i+1})$ is for the relative mean poses between adjacent time steps. The covariance between adjacent time steps is $\widetilde{\Sigma}_{i,i+1} = Ad_{i,i+1}\Sigma_{i,i+1}Ad_{i,i+1}^T$ [25].

E. Adaptation to novel situations

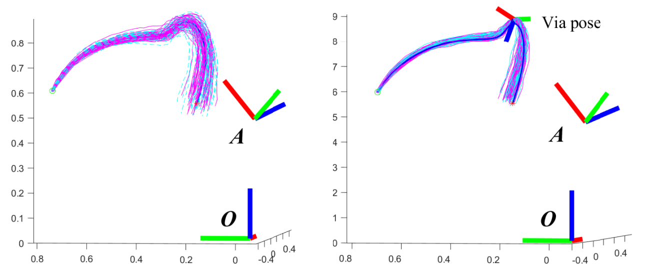
One of the most essential abilities of an LfD method is its adaptability to novel and unseen situations.

1) Adaptation to via points with uncertainties: Suppose that the robot is asked to pass a via point $g_i^* \in$ SE(3) with uncertainty, which is described by covariance matrix Σ_i^*. The posterior distribution, as shown in Figure 2, can be computed using an observation mode.



(a) Original trajectory mean and samples (b) Adaptation to a new goal pose (c) Adaptation to new via points

Fig. 2: Examples of the adaptation to new via points with uncertainties. The solid blue and magenta curves are the mean of the encoded joint prior and posterior distribution, respectively. Dashed magenta curves are the random trajectory samples from the probability distribution.

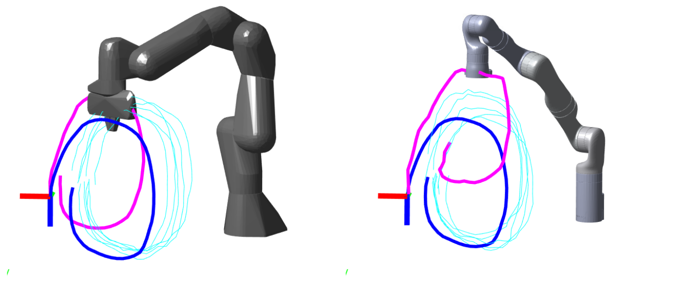


(a) Encoded joint distribution. (b) Distribution after conditioning on a via point.

Fig. 3: The 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 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).

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 $g^o = h^{-1}gh$ [26]. The conditional probability between two adjacent frames after the change of view can be computed as $\boldsymbol{x}_i^o = \log^\vee(h^{-1}\mu_i^{-1}g_i h) = Ad^{-1}(h)\boldsymbol{x}_i$. The distribution of the whole trajectory as viewed in frame A can be obtained by the equivariance property under the change of view.

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 [9], [27]. Workspace density can be approximated by the convolution of Gaussian distribution on SE(3) [28]. In this work, the workspace density is used to inform the learned trajectory distribution, to stay closer to the higher probability region within the robot workspace. The distribution of each intermediate pose along the trajectory is conditioned by this density function, which can be viewed as a fully observable model. The posterior probability distribution based on the respective workspace density of Franka Emika Panda and Kinova Gen3 robots is shown in Figure 4.



(a) Franka Emika Panda (b) Kinova Gen3

Fig. 4: Fusion with robot-specific workspace density. Thin cyan curves are the demonstrated trajectories using 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.

III. 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 $g_r^{(k)} = \left(R_r^{(k)}, \mathbf{t}_r^{(k)}\right)$. And the cost function $\mathbf{c}(\mathbf{q}_i, t_i)$ for i^{th} time step is computed as

$$\begin{aligned}\mathbf{c}(\mathbf{q}_i,t_i) =& \frac{1}{m_\mathrm{r}}\sum_{k=1}^{m_\mathrm{r}}\Big(w_\mathrm{rot}\left\|\log^\vee\left(R^T(\mathbf{q}_i,t_i)\,R_\mathrm{r}^{(k)}(t_i)\right)\right\| \\ &+ w_\mathrm{tran}\left\|\mathbf{t}(\mathbf{q}_i,t_i)-\mathbf{t}_\mathrm{r}^{(k)}(t_i)\right\|\Big)\end{aligned} \quad . \tag{6}$$

Algorithm 2: Cost function for Workspace-STOMP based on trajectory distribution

```
Inputs      : q: Rollout joint angles;
              $\left\{ g_{\mathrm{r}}^{(k)}(t) \right\}$: Sampled trajectories from the
  distribution learned by PRIMP
Parameters: $w_{\mathrm{rot}}$, $w_{\mathrm{tran}}$: Weights for rotation and
              translation parts in the distance function
Outputs     : $\mathbf{c}(\mathbf{q}, t)$: Cost value for each rollout
1 for time step $i$ do
2     $g(\mathbf{q}_i, t_i)$ = ForwardKinematics($\mathbf{q}_i$);
3     Compute $\mathbf{c}(\mathbf{q}_i, t_i)$ using Eq. (6) ;
4 end
```

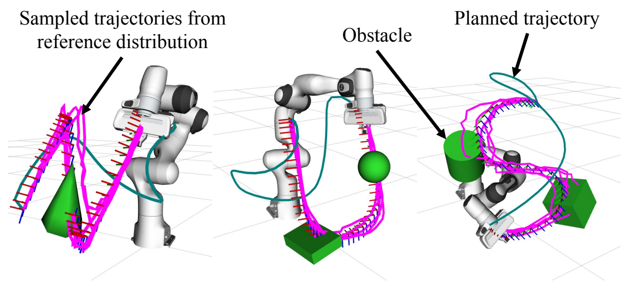


Fig. 5: Motion planning using Workspace-STOMP to follow the learned trajectory distribution from PRIMP.

The weights w_{rot} and w_{tran} for rotation and translation parts in the distance function are set by users. The computational process is shown in Alg. 2.

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 [29]. Simulations of writing letters "N", "U" and "S" using the proposed Workspace-STOMP are shown in Figure 5.

IV. Integrated Robotic System

The proposed motion primitives learning and motion planning methods are integrated into a robotic system, with other components including environment perception, tasks prior knowledge, and key pose detection by either marker or robot imagination for unseen objects.

A. Workflow

The workflow of the system is shown in Figure 6. 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 trajectory probability distribution generation by PRIMP. For a new planning request, the new scenario is obtained by 3D reconstructions using an RGB+D camera. Then *key poses* for the task are generated based on task prior knowledge, as shown in Table. I. A set of key pose candidates are then fed into PRIMP to condition the trajectory probabilistic distribution. The posterior distribution is used to guide the STOMP planner with new planning scenes,

TABLE I: Prior knowledge of key pose generation methods for different tasks.

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

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 re-planning.

B. Task prior knowledge

Key pose generation methods are defined by the operator and stored as the prior knowledge for different tasks, as listed in Table I. Key poses are defined for the end-effector and can be viewed as constraints for a task to be successfully fulfilled, i.e., the pouring task needs to be informed of the target pouring pose that varies based on the shape and location of a container. The number and time steps of key poses for each task may differ, i.e., the scooping task needs the scooping pose to fulfill the scoop action, and the scooping pose is assumed to be in the middle during scooping.

Key poses can be obtained by two methods, either by reading ArUco tags or robot imagination. The former one is used for tasks including transporting and door opening. For door opening, the goal pose is further subjected to the geometry constraint of the door. The latter method is useful for tasks like pouring and scooping, where the key poses are conditioned based on the shape and pose of the object. Instead of defining priori all possible key pose candidates for each object, this work used robot imagination, which detects object affordance through physical simulation with pre-defined motions and finds object functional poses for later robot manipulation. Robot imagination allows extending skills to unseen objects. This is first proposed for the pouring affordance for unseen containers in [19]. This work proposed scooping imagination, as shown in Figure 7, which has a similar pipeline as the pouring imagination to find scooping pose candidates. The reconstructed mesh of the object is first loaded into the simulation environment, and red particles are dropped into the object based on the location and size of its bounding box. The spoon model is used for conducting the pre-defined scooping action. Each time, the initial pose of the spoon is sampled based on the bounding box of the object. The scooping pose at time step $t = 0.5$ depends on

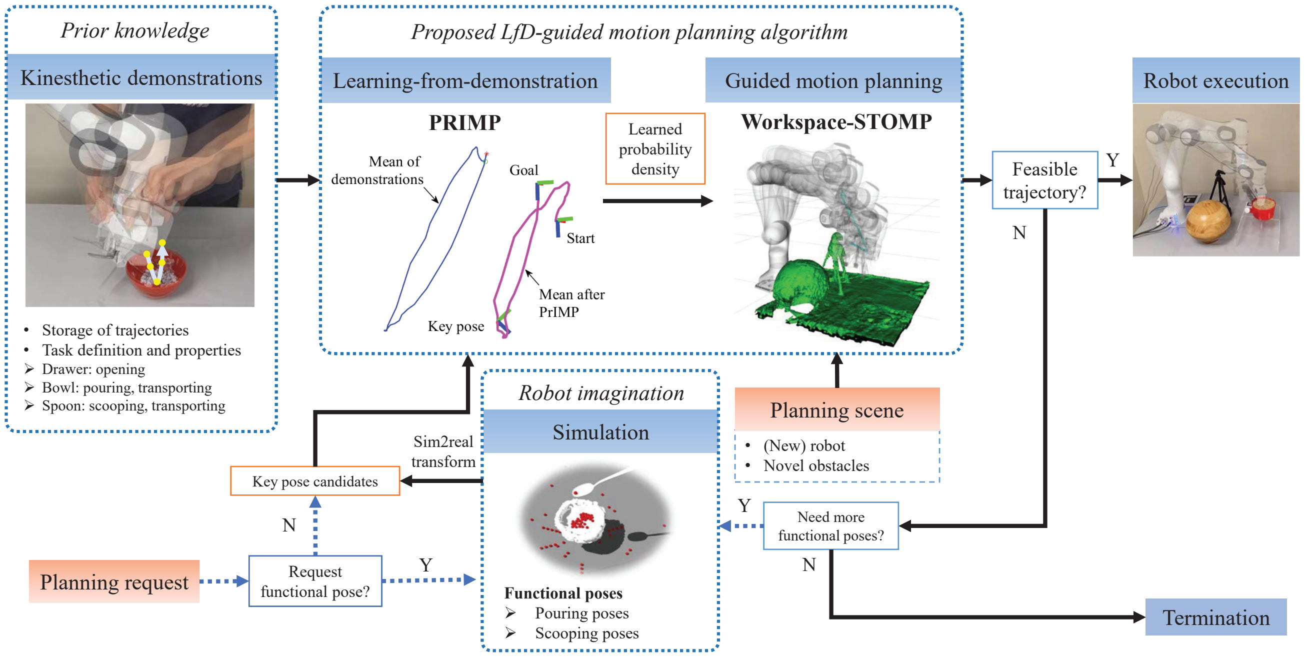


Fig. 6: Workflow of the proposed robotic system including PRIMP, Workspace-STOMP and robot imagination.

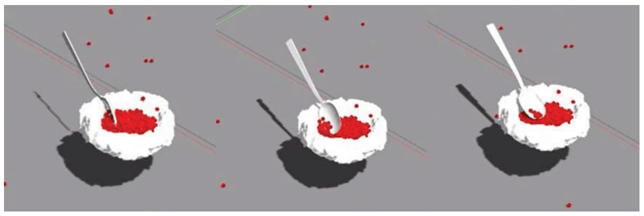


Fig. 7: The robot imagination process for scooping. The white meshed object represents the container that is unseen during demonstration.

the position of the lowest particle among particles contained by the object. If the spoon successfully scoops any particle, the scooping pose is recorded as a possible key pose for the object. Both pouring and scooping simulations were conducted in the Gazebo physics simulator.

V. Physical Experiments and Evaluation

Physical experiments using the Franka Emika Panda robot are conducted. Human operator demonstrates each skill by dragging the end-effector for 5-10 trials (Fig. 8)

A. Benchmarks among learning-from-demonstration methods

The proposed PRIMP method is compared with ProMP [6] and Orientation-KMP [8], both probabilistic LfD methods. For Orientation-KMP, different levels of magnitude for the kernel parameter are varied. With manually defining 50 different pairs of goals and via poses, different LfD algorithms are applied to re-produce the task. The results are shown in Table II and Table III. Table II evaluates the similarity between demonstrated trajectories and learned trajectories in terms of distances of

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

rotation and translation parts. Table III compares the distance between the designated via point μ_i^* and learned poses g_i sampled from the posterior trajectory distribution.

Table II shows that PRIMP learns the core features from the demonstration for both orientation and translation, *i.e.*, Task 1 and 5 require objects rotating along a fixed axis in the space, and PRIMP learns this core features while KMP fails. Table III demonstrates the ability to adapt the learned trajectory distribution by PRIMP to new via points, especially the rotation part. Benchmark results show that PRIMP outperforms other probabilistic methods in terms of similarity metric for the whole trajectory as well as the distance to the desired via point.

B. Comparisons on guided motion planning

Benchmarks for motion planning are conducted, which include three manually defined environments in simulation for the 5 daily tasks. Environment types include sparse, cluttered, and dense, depending on the difficulty of motion planning. We compare the proposed Workspace-STOMP planner with (1) STOMP [17] without guidance and (2) Cartesian-guided

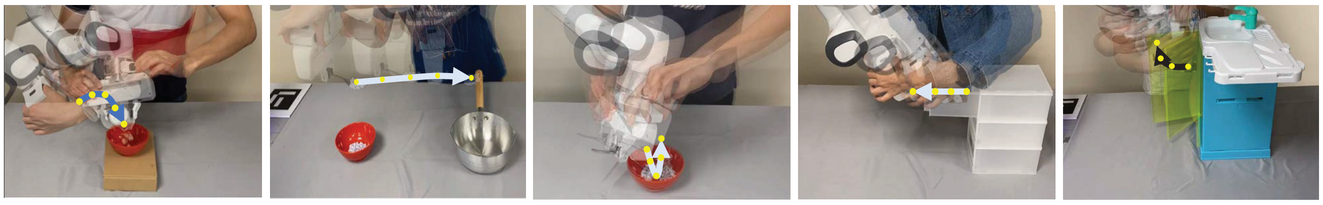

Task 1: Pouring Task 2: Transporting Task 3: Scooping Task 4: Drawer Opening Task 5: Door Opening

Fig. 8: Kinesthetic demonstrations for different tasks.

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: 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

STOMP [16]. 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 IV shows the *expected* task solving time by considering the task success rate [30], *i.e.*,

$$\mathcal{E}_{\text{plan}} \doteq \frac{\overline{T}_{\text{plan}}}{\rho_{\text{task}}}, \tag{7}$$

where $\overline{T}_{\text{plan}}$ is the averaged planning time among all the trials. This metric provides a trade-off between the planning time and success rate, *i.e.*, a method for a task is desirable if it can solve in a short time with high success rate. Even if the method is able to solve the problem for many trials, but have to spend quite a long time, the expected time might still be long. When all the trials are failed, 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 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 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 some trials.

VI. Conclusion

This article presents *PRobabilistically-Informed Motion Primitives (PRIMP)*, a learning-from-demonstration 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. For future works, the proposed skills learning and motion planning framework can be extended to more applications, *i.e.*, object flipping or stacking. The learned motion primitives can also be combined with high-level task planning for long-horizon manipulations.

Acknowledgement

The authors would like to thank Dr. Seng-Beng Ho and Mr. Jikai Ye for useful discussions. This work was supported by NUS Startup grants A-0009059-02-00, A-0009059-03-00, CDE Board account E-465-00-0009-01, National Research Foundation, Singapore, under its Medium Sized Centre Programme - Centre for Advanced Robotics Technology Innovation (CARTIN), sub award A-0009428-08-00 and AME Programmatic Fund Project MARIO A-0008449-01-00. The ideas expressed in this paper 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," *Annual review of control, robotics, and autonomous systems*, vol. 3, pp. 297–330, 2020.

[2] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Springer handbook of robotics*. Springer, 2008, pp. 1371–1394.

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

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

[5] J. Kober and J. Peters, "Policy search for motor primitives in robotics," *Advances in neural information processing systems*, vol. 21, 2008.

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

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

[8] Y. Huang, F. J. Abu-Dakka, J. Silvério, and D. G. Caldwell, "Toward orientation learning and adaptation in Cartesian space," *IEEE Transactions on Robotics*, vol. 37, no. 1, pp. 82–98, 2020.

[9] G. S. Chirikjian and A. B. Kyatkin, *Harmonic analysis for engineers and applied scientists: updated and expanded edition*. Courier Dover Publications, 2016.

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

[11] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Robot placement based on reachability inversion," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 1970–1975.

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

[13] D. Koert, G. Maeda, R. Lioutikov, G. Neumann, and J. Peters, "Demonstration based trajectory optimization for generalizable robot motions," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 515–522.

[14] G. Ye and R. Alterovitz, "Guided motion planning," in *Robotics research*. Springer, 2017, pp. 291–307.

[15] B. Magyar, N. Tsiogkas, B. Brito, M. Patel, D. Lane, and S. Wang, "Guided stochastic optimization for motion planning," *Frontiers in Robotics and AI*, p. 105, 2019.

[16] M. Dobiš, M. Dekan, A. Sojka, P. Beňo, and F. Duchoň, "Cartesian constrained stochastic trajectory optimization for motion planning," *Applied Sciences*, vol. 11, no. 24, p. 11712, 2021.

[17] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 4569–4574.

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

[19] H. Wu and G. S. Chirikjian, "Can I pour into it? Robot imagining open containability affordance of previously unseen objects via physical simulations," *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 271–278, 2020.

[20] V. Vosylius and E. Johns, "Where to start? Transferring simple skills to complex environments," in *6th Annual Conference on Robot Learning*, 2022.

[21] W. Jin, T. D. Murphey, D. Kulić, N. Ezer, and S. Mou, "Learning from sparse demonstrations," *IEEE Transactions on Robotics*, 2022.

[22] T. W. Mitchel, S. Ruan, and G. S. Chirikjian, "Signal alignment for humanoid skeletons via the globally optimal reparameterization algorithm," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2018, pp. 217–223.

[23] G. S. Chirikjian, "Bootstrapping globally optimal variational calculus solutions," *Calculus of Variations and Partial Differential Equations*, vol. 62, no. 2, pp. 1–33, 2023.

[24] M. K. Ackerman and G. S. Chirikjian, "A probabilistic solution to the AX=XB problem: Sensor calibration without correspondence," in *International Conference on Geometric Science of Information*. Springer, 2013, pp. 693–701.

[25] G. S. Chirikjian, *Stochastic models, information theory, and Lie groups, volume 2: Analytic methods and modern applications*. Springer Science & Business Media, 2011, vol. 2.

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

[27] Y. Wang and G. S. Chirikjian, "Nonparametric second-order theory of error propagation on motion groups," *The International journal of robotics research*, vol. 27, no. 11-12, pp. 1258–1273, 2008.

[28] ——, "Workspace generation of hyper-redundant manipulators as a diffusion process on SE(N)," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 3, pp. 399–408, 2004.

[29] S. Chitta, I. Sucan, and S. Cousins, "Moveit! [ROS topics]," *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 18–19, 2012.

[30] J.-M. Lien, "Hybrid motion planning using minkowski sums," *Proceedings of robotics: science and systems IV*, 2008.