Generalization of Task Parameterized Dynamical Systems using Gaussian Process Transportation

Giovanni Franzese, Ravi Prakash, and Jens Kober

Abstract*—Learning from Interactive Demonstrations has revolutionized the way non-expert humans teach robots. It is enough to kinesthetically move the robot around to teach pick-and-place, dressing, or cleaning policies. However, the main challenge is correctly generalizing to novel situations, e.g., different surfaces to clean or different arm postures to dress. This article proposes a novel task parameterization and generalization to transport the original robot policy, i.e., position, velocity, orientation, and stiffness. Unlike the state of the art, only a set of points are tracked during the demonstration and the execution, e.g., a point cloud of the surface to clean. We then propose to fit a non-linear transformation that would deform the space and then the original policy using the paired source and target point sets. The use of function approximators like Gaussian Processes allows us to generalize, or transport, the policy from every space location while estimating the uncertainty of the resulting policy due to the limited points in the task parameterization point set and the reduced number of demonstrations. We compare the algorithm's performance with state-of-the-art task parameterization alternatives and analyze the effect of different function approximators. We also validated the algorithm on robot manipulation tasks, i.e., different posture arm dressing, different location product reshelving, and different shape surface cleaning. A video of the experiments can be found here: `https://youtu.be/FDmWF7K15KU`.

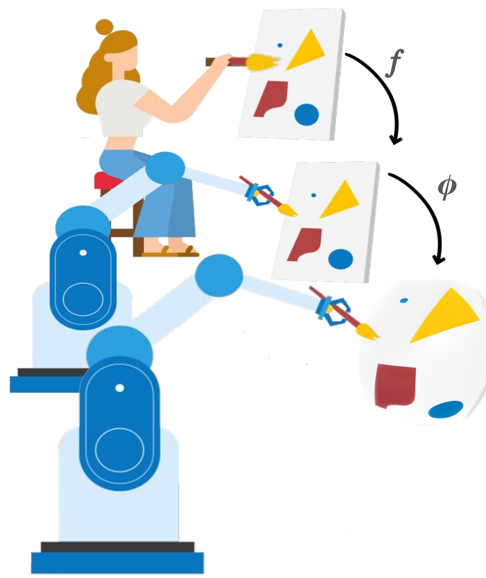

Fig. 1: Example of Policy Transportation. The human demonstrates to a robot how to perform a task on a flat canvas. Then, the robot, when facing a new curved canvas, "transports" its knowledge in the new situation by adapting the end effector velocity, orientation, and stiffness to correctly adapt the drawing on the new canvas.

I. Introduction

One of the main appeals of robot learning from demonstrations is that it enables humans with different levels of robotic expertise to transfer their knowledge and experience about skills and tasks to the robot [1]. This alleviates the need to program such skills by hand, which is tedious, error-prone, and requires an expert. However, one of the long-term challenges of this approach is generalizing the learned behavior to novel situations.

By enhancing the policy with task parameterization [2], robots can generalize their learned knowledge to different variations of the same task, thus promoting scalability and data efficiency in robot learning, allowing robots to learn faster and adapt to new scenarios. For instance, a robot can be trained to clean surfaces with a reduced set of shapes, to dress an arm in a certain configuration, or to pick objects with a certain shape and place them on the right shelf. Ideally, the robot would generalize the learned skill to novel situations without extensive retraining. This paper proposes a way of transfer, or "transport", the original learned behavior from the old to the new situation.

For example, let us imagine teaching a robot how to draw on a flat canvas, as depicted in Fig. 1. The robot can learn to imitate the human on the same flat canvas, however, when the desired surface to reproduce the task is changed, i.e., the blue curvy surface, the user may have to teach the task again on the new surface. Instead, by knowing the correspondence of a set of points from the two surfaces, we propose to learn a function that could generalize the policy from the original parameter distribution to the new one.

The learned function is locally deforming the space according to the new location of the tracked points and this can be used to generalize the learned velocity field, learned stiffness and orientation. Fig. 2, depicts how the trajectory and the learned desired dynamics of drawing a letter C depicted as red dots, on a flat surface, depicted in green; the robot would learn the desired behavior, i.e., the velocity depicted as black arrows, form any position of the space. However, when the desired surface to reproduce the task is changed, i.e., the blue curvy surface, the user may have to teach the task again on the new surface. Instead, by knowing the correspondence of a set of points from the two surfaces, we show how to learn a function that could generalize the policy from the original surface shape to the new one.

Our algorithm contributes with the formalization and testing of a policy transportation theory that can

This research is funded by European Research Council Starting Grant TERI "Teaching Robots Interactively", project reference 804907.

The authors are with the Cognitive Robotics, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail:g.franzese@tudelft.nl; r.prakash-1@tudelft.nl: j.kober@tudelft.nl).

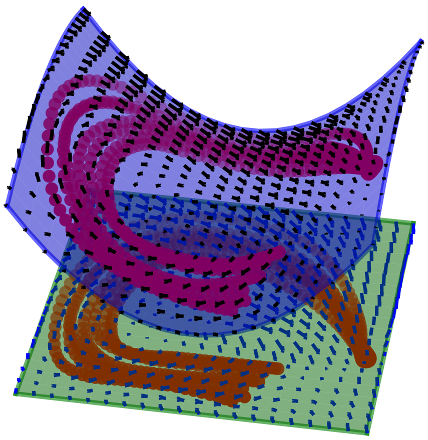

Fig. 2: The demonstrations (in red) are given on the green surface, and the learned dynamics, depicted as arrows, are learned from them. Later, the demonstration and the dynamics are projected on another curved blue surface using the proposed Gaussian Process Transportation.

- transport the demonstration from the original space to the new space, see Sec. IV-B;
- transport the velocity field, end-effector orientations, stiffness and damping by exploiting the derivative of the transportation mapping, see Sec. IV-C;
- estimates the final uncertainty due to the reduced set of demonstrations and the estimated uncertainty in the transportation map, see Sec. IV-E.

The same algorithm was tested on generalizing complex manipulation tasks like cleaning surfaces with different shapes, picking and placing objects at other locations, and dressing an arm in various configurations, see Sec. VI.

II. Related Works

One classical method to generalize behavior to new situations involves task parameterization, such as the picked object location, target goal, or via points. This idea of behavior representation and generalization in varying task configurations has been popularly achieved using Dynamic Movement Primitives (DMPs) [3] through a single or multiple demonstration per task. The DMP model consists of stable second-order linear attractor dynamics with alterable target parameters (end goal or velocities).

An approach to adapt the DMPs via points is addressed in [4], but it demands combining several DMPs for a single task. Alternatively, a roto-translation can be applied to the original dynamical systems according to the tracked frame or points in the environment.

Approaches for modeling and generalizing demonstrations that have shown improved performances with respect to the DMPs are Probabilistic Movement Primitives (ProMPs) [5]. ProMPs model the distribution over the demonstrations that capture temporal correlation and correlations between the DoFs using a linear combination of weights and a set of manually designed basis functions. Adaptation to new task parameters or via points is achieved using Gaussian conditioning. While this approach allows modeling the structure and variance of the observed data in the absolute reference frame, the generalization to the new task parameters is satisfactory only within the confidence bound of the demonstration data. For example, showing many demonstrations for different goal points, the probabilistic model can be conditioned on a novel object position and retrieve the most probable trajectory that brings the robot to that final position.

However, when learning reactive policies, i.e., a function of the state and not of the phase of the motion, the use of ProMPs is limited since the number of basis functions overgrows with the dimension of the input, limiting its applications.

Kernelized Movement Primitives (KMP) [6] proposed a non-parametric skill learning formulation.

This formulation allows modulation of the recorded trajectories to new via points, obtaining the deformation of the original movement primitives given the temporal correlation of the demonstration and the via point, calculated with the kernel function. However, the user must specify the time and the corresponding waypoint to deform the original trajectory or rely on a heuristic that, for example, matches each waypoint with the closest point in the trajectory.

Gaussian Mixture Models (GMM) [7]–[9] have successfully been employed in modeling demonstration, endowing with a successful generalization in its task parameterized version (TP-GMM) [2].

Given a set of reference frames that are tracked during demonstration and execution, the central idea of TP-GMM is the local projection of the demonstrations in each of the local reference frames and encoding each model as a mixture of Gaussians [2], [10]–[12]. The local models are then fused in global coordinates, using the Product of Gaussian (PoG), and a new motion is rolled out from the resulting mixture model. This approach, however, requires many demonstrations to fit the model and does not scale well with the increasing number of task frames. This is because the PoG does not scale well when dealing with many reference frames and can lead to undesirable generalizations.

The generalization of the demonstrations, encoded as a chain of events in a graph, with respect to multiple via points, can be done using Laplacian Editing [13], [14]. It uses Laplace-Betromi operator, a well-known algorithm in the computer graphics community, to deform meshes [15], to encode geometric trajectory properties and generate deformed trajectories using task constraints, i.e., new via points. The operator ensures a smooth deformation of the trajectory through the via points.

However, this approach is very specific for trajectory reshaping and requires explicit knowledge of the new via-point for some trajectory nodes.

In this paper, we relax the need to explicitly specify the new via-points for a specific node in the trajectory since the via-point definition can be error-prone and requires ad-hoc algorithms.

Moreover, while all the previous approaches only address generalizing a demonstration/robot trajectory, our proposed approach relaxes the requirement of generalizing only trajectories. It provides the means also to generalize the velocity field of the original dynamics, as well as the orientation and the stiffness.

For generalization, we use a combination of linear transformation and non-linear deformation to transport demonstrations to new situations while estimating the process uncertainty. We validated the algorithm on three use cases: robot reshelving, robot dressing, and robot cleaning.

Robot Reshelving Authors of [16] propose, within the realm of robotic retail automation, to enable non-expert supermarket employees to teach a robot a reshelving task and then adeptly generalize its learned policy to accommodate diverse task situations. The generalization of the policy for varying object locations is achieved by switching between the dynamical system learned between the object and the goal frame. However, the switching strategy entails having a good prior on when to switch and all the possible implications of generating instability by suddenly changing the policy online. TP-GMM alternatives [2], solved the problem of the switching by obtaining the final GMM as the product of the relative models; however more than one demonstration is necessary to fit an informative model.

Robot Dressing Robot dressing is a challenging task since it includes manipulation of deformable objects, and the margin of error to correctly go through the human arm is very low.

Task parameterized dynamical system has been applied to learn the dressing task in the robotics research. The dressing demonstrations w.r.t. the wrist and the shoulder of a human arm have been used to learn a dressing policy via DMP [17], TP-HMM [18] and a TP-GMM [19].

Robot Surface Cleaning Efficient and fast generalization of robotic surface cleaning is achieved using task-parameterized learning.

In [20] the cleaning dynamics is the sum of two dynamical systems, one that learns the desired motion on the surface and another that computes the modulation term on the desired force to apply on the perpendicular direction of the surface (where the shape is known a priori). This second term is learned as a non-linear function that allows learning larger forces in a region of the surface compared to others.

The shape of the surface can also be estimated using the wrench measured with a force-torque sensor attached at the end-effector; for instance, [21] generalizes the polishing task on the novel curved surface by adapting the orientation and the direction of the contact force such that to minimize perceived torque.

The following section will provide some background on the main concepts necessary for learning a policy from demonstration and generalize it using Gaussian Process Transportation.

III. Preliminaries

We review the learning of policies from demonstration using a dynamical system formulation in Sec. III-A and the specific use of a Gaussian Process (GP) for learning the policy itself or the transportation policy in Sec. III-B.

A. Interactive Learning of Dynamical System from Demonstrations

A non-linear dynamical system (DS) can be described by

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) \tag{1}$$

where $\boldsymbol{x}$ is the state of the system, such as the end effector or the joint position, while $\dot{\boldsymbol{x}}$ is the observed state transition.

The demonstrated dynamics are captured by a non-linear function f that is agnostic to the function approximation used. However, the resulting properties for out-of-distribution predictions depend on it. For example, the robot may extrapolate the wrong behavior and make the motion to diverge in undesired regions of the workspace. To enforce stability of the motion dynamics, the Stable Estimator of Dynamical Systems (SEDS) [22], constrains the properties of each of the linear models of a Gaussian mixture to enforce global asymptotic stability. Alternatively, the global dynamics can be coupled with latent (linear) stable dynamics, using a deep neural network projector to ensure the global stability of the learned task dynamics [23], [24].

Differently, [25] proposes to minimize the epistemic uncertainties predicted as an output of a GP dynamical system to attract the motion in regions with high confidence, i.e., close to the training set. Moreover, [25] also proposes a way of updating the learned attractor and stiffness field online, to learn for example, how to apply enough pressure on a table, and clean it successfully. Fig. 2 shows the dynamics learned from the drawing of a letter C from multiple demonstrations, where the attractor delta in z is set to be constantly negative, i.e., the vector arrows point under the surface. Since, in the context of this article, the policy and the transportation are learned with a Gaussian Process, the next section will introduce the fundamental equations used to fit and predict mean and variance using a GP (or its approximation).

B. Gaussian Process Regression

A Gaussian Process is a collection of random variables, any finite number of which have a joint Gaussian distribution. To fit a Gaussian process, we start with a prior distribution over functions. The prior is typically specified as a mean function and a kernel function. The prior distribution represents our beliefs about the functions before observing any data. For example, when learning a dynamical system, it is safer to have a zero mean prior, such that the robot does not attempt to do any movement if there is no significant evidence from the human demonstration.

The posterior distribution is used to make predictions or perform inferences, and assuming Gaussian likelihood, the mean and variance predictions are

$$\boldsymbol{\mu} = \boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}}(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{y} \tag{2}$$

$$\boldsymbol{\Sigma} = \boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}_*} - \boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}}(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}_*} \tag{3}$$

where $\boldsymbol{X}_* \in \mathbb{R}^{n\times m}$ are the evaluation inputs and $\boldsymbol{X} \in \mathbb{R}^{N\times m}$ and $\boldsymbol{y} \in \mathbb{R}^{N\times l}$ are the training inputs and outputs; the correlations $\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} \in \mathbb{R}^{N\times N}$, $\boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}} \in \mathbb{R}^{n\times N}$, $\boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}_*} \in \mathbb{R}^{n\times n}$ are the correlation matrix between every input training point, between testing and training points and between testing points.

Every entry of this matrices is computed using a kernel function, for example, the squared exponential kernel, defined as

$$k_{\mathrm{SE}}(\boldsymbol{x_i}, \boldsymbol{x_j}) = \sigma_p^2 \exp\left(-\frac{(\boldsymbol{x_i} - \boldsymbol{x_j})^2}{2\ell^2}\right),$$

where $\boldsymbol{x_i}, \boldsymbol{x_j}$ are any pair of training-training, testing-training, or testing-testing data points; σ_p and ℓ are the kernel hyperparameters and, together with the likelihood noise σ_n are optimized with a maximization of the log-likelihood of the given data [26].

Given the computationally expensive inversion of the covariance matrix that grows with the number of data points, optimizing kernel hyperparameters and inference becomes prohibitive when having big datasets. Therefore, variational inference aims to approximate the true posterior $p(\boldsymbol{f}|\boldsymbol{X}, \boldsymbol{y})$ with a simpler distribution $q(\boldsymbol{f})$. Stochastic Variational Gaussian Process (SVGP) optimizes the location and values of a set of pseudo data, usually referred to as inducing points $\boldsymbol{Z}$ and a variational distribution $q(\boldsymbol{u})$. The parameters of the kernel and the variational parameter of the approximated distribution are optimized using the evidence lower bound (ELBO) as the evidence of the data [27].

The use of SVGP, in the context of this paper, can make the computation significantly faster when fitting a transportation function with many points from the source and target distribution; however, any of the proposed algorithms, formalized in the next section, is independent of the approximation nature of the GP.

IV. Policy Transportation

We learn and correct a manipulation policy from an interactive demonstration [1], using, for example, a GP as the function approximator to fit the demonstration dynamics.

However, although the task execution would succeed from any given starting configuration of the robot, it will fail to generalize if the task is changed, e.g., if the object to pick is moved or if the robot faces a differently shaped surface to clean.

Intending to find a task parameterization that scales from pick-and-place to continuous surface, we assume to track a set of environment-specific points that are descriptive of the situation. For example, when picking a box, the eight corners are tracked, or when cleaning a surface, a point cloud representation is used. This is the most straightforward yet most general task parameterization while being technically feasible, given the current development of LiDAR and depth camera technology.

We define the tracked N point recorded in the demonstration scenario as the source distribution, i.e.,

$$\boldsymbol{\mathcal{S}} = \{(x_{s,i}, y_{s,i}, z_{s,i})\}_{i=1}^{N} \tag{4}$$

while the moved points in the new scenario are defined as the target distribution, i.e.,

$$\boldsymbol{\mathcal{T}} = \{(x_{t,i}, y_{t,i}, z_{t,i})\}_{i=1}^{N}. \tag{5}$$

We assume that the points of target and source distribution are already paired. Many algorithms are available to (optimally) pair the two distributions [28]; hence this is not the focus of our work.

We define a map $\boldsymbol{\phi}$ such that each point s_i in S is paired with one and only one point t_j in T. This can be represented as:

$$\begin{aligned} \boldsymbol{\phi} : \boldsymbol{\mathcal{S}} &\to \boldsymbol{\mathcal{T}} \\ s_i &\mapsto t_j \end{aligned} \tag{6}$$

where $i, j \in \{1, 2, \ldots, n\}$. We aim to find the function that maps from the source space to the target space, given the evidence of the input-output pairs from the source and target distribution. Estimating a continuous process allows the deformation of the complete space to match the source and the target distribution, as depicted in Fig. 3.

The structure of the function $\boldsymbol{\phi}$ that we want to approximate can be any nonlinear function that maps any point of the Cartesian space to itself, e.g., $\boldsymbol{\phi} : \mathbb{R}^3 \to \mathbb{R}^3$. However, in the context of this article, we consider the transportation function to have the following definition,

$$\boldsymbol{\phi}(\boldsymbol{x}) := \boldsymbol{\gamma}(\boldsymbol{x}) + \boldsymbol{\psi}(\boldsymbol{\gamma}(\boldsymbol{x})) \tag{7}$$

where $\boldsymbol{\gamma}$ and $\boldsymbol{\psi}$ are, respectively, a linear and nonlinear transformation. The fitting of the function is made in two steps: first the linear transformation $\boldsymbol{\gamma}(\boldsymbol{x})$ is obtained, and then the nonlinear transformation $\boldsymbol{\psi}(\boldsymbol{\gamma}(\boldsymbol{x}))$ is fitted on the residual error.

A. Linear Transformation

To fit the optimal rotation matrix between the source and the target distribution, the centered source and target distribution are used as labels for the fitting of the function $\boldsymbol{\gamma}$, i.e.

$$\underbrace{\boldsymbol{\mathcal{T}} - \bar{\boldsymbol{\mathcal{T}}}}_{y_{\text{label}}} = \boldsymbol{\gamma}(\underbrace{\boldsymbol{\mathcal{S}} - \bar{\boldsymbol{\mathcal{S}}}}_{x_{\text{label}}}) \tag{8}$$

where $\bar{\boldsymbol{\mathcal{S}}}$ and $\bar{\boldsymbol{\mathcal{T}}}$ are the centroid of the source and the target distribution, respectively. We can find the rotation between the two centered distributions using the Singular Value Decomposition (SVD) imposing

$$\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V}^\top = (\boldsymbol{\mathcal{S}} - \bar{\boldsymbol{\mathcal{S}}})^\top (\boldsymbol{\mathcal{T}} - \bar{\boldsymbol{\mathcal{T}}}) \tag{9}$$

and the rotation matrix is defined as,

$$\boldsymbol{A} = \boldsymbol{V}\boldsymbol{U}^\top, \tag{10}$$

however, if $\det(\boldsymbol{A}) < 0$, the last column of $\boldsymbol{V}$ is flipped in sign, and the computation of the rotation matrix is repeated. This ensures the transformation is a proper rotation matrix without any reflection; see [29] for more details. Hence, the linear transformation on any point in the space can be computed as

$$\boldsymbol{\gamma}(\boldsymbol{x}) = \boldsymbol{A}(\boldsymbol{x} - \bar{\boldsymbol{\mathcal{S}}}) + \bar{\boldsymbol{\mathcal{T}}}. \tag{11}$$

Fig. 3,c shows a linear transformation of the source and a grid of points from the original space depicted in Fig. 3,b.

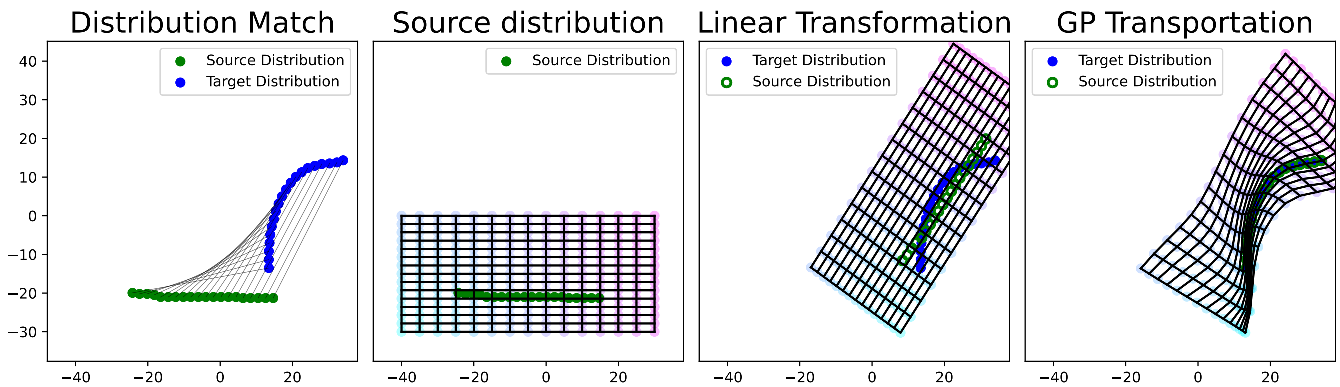

Fig. 3: 2D transportation. **Distribution match** depicts the source and the target distribution correspondence used to train the transportation function. **Source Distribution** depicts a grid of points in the original space. **Linear Transformation** shows the effect on the original grid when only a linear transformation is used to match source and target distribution. **GP Transportation** captures the deformation of the space when the source points are forced to match the target ones.

B. Non-linear Transportation

After fitting the linear transformation of Eq. (11), the residual transformation is obtained by substituting the source, target points, and the fitted linear function in Eq. (7), obtaining that

$$\underbrace{\boldsymbol{\mathcal{T}} - \gamma(\boldsymbol{\mathcal{S}})}_{y_{\text{label}}} = \boldsymbol{\psi}(\underbrace{\gamma(\boldsymbol{\mathcal{S}})}_{x_{\text{label}}}). \tag{12}$$

The nonlinear function $\boldsymbol{\psi}$ can be any nonlinear regressor, such as a Neural Network, a Random Forest, a Gaussian Process, etc. However, the inducting bias given by the nature of the nonlinear function will affect the regression output when going out of distribution, i.e., far away from the given data. For example, suppose the function is approximate with a GP with a distance-based kernel k, such as a square exponential kernel.

If the prior distribution is set to be a zero-mean function, when making predictions, see (2) in regions of the space far away from the source distribution points, the final transportation converges to just being a linear transformation, see Fig. 3,d. Knowing the out-of-distribution (o.o.d.) properties of our transportation policy is desirable, considering that we will transport points that are not necessarily close to the point of the source/target distribution.

C. Transportation of the Dynamics

Although the transportation map allows the transport of any point of the original demonstration in the new situation, for example, to generalize the demo on cleaning a new surface, we still have not formulated a transportation function for the velocity field $\dot{\boldsymbol{x}}$. It is not as trivial as computing the numerical differentiation of the transported trajectories. We consider the policy labels as independent points, no longer part of a trajectory. This allows us to learn from multiple demonstrations and to change the velocity label connected to them if providing (teleoperated) feedback or aggregating new data from interactive demonstrations [25].

Nevertheless, the partial derivative of the transportation mapping can be exploited in the velocity field generalization. Given the transportation function defined in the source space and projecting in the target space, i.e.,

$$\hat{\boldsymbol{x}} = \boldsymbol{\phi}(\boldsymbol{x})$$

by differentiating w.r.t. time on both sides and using the chain rule, we obtain the velocity field in the transported space as

$$\dot{\hat{\boldsymbol{x}}} = \frac{\partial \boldsymbol{\phi}(\boldsymbol{x})}{\partial \boldsymbol{x}} \dot{\boldsymbol{x}} = \boldsymbol{J}(\boldsymbol{x}) \dot{\boldsymbol{x}} \tag{13}$$

where the Jacobian matrix, using the definition of Eq. (7), can be defined as

$$\boldsymbol{J}(\boldsymbol{x}) := \frac{\partial \gamma(\boldsymbol{x})}{\partial \boldsymbol{x}} + \frac{\partial \boldsymbol{\psi}(\boldsymbol{x})}{\partial \gamma(\boldsymbol{x})} \frac{\partial \gamma(\boldsymbol{x})}{\partial \boldsymbol{x}}$$

where $\frac{\partial \gamma(\boldsymbol{x})}{\partial \boldsymbol{x}} = \boldsymbol{A}$ and $\frac{\partial \boldsymbol{\psi}(\boldsymbol{x})}{\partial \gamma(\boldsymbol{x})}$ can be obtained using automatic differentiation of the chosen regressor. In the following sections, we will simplify notation by omitting the explicit dependence of $\boldsymbol{J}$ on $\boldsymbol{x}$.

Fig. 4 summarizes all the steps done during transportation that can also be visualized in Fig. 5 on a 2D example. In the first row of the two figures, the original demonstration is used to compute the vector field. Then, in the second row, the demonstration and the vector field are transported, thanks to the transportation map $\boldsymbol{\phi}$ fitted on the observed source and target distribution.

In particular, after the transportation of $\hat{\boldsymbol{x}}$ and $\dot{\hat{\boldsymbol{x}}}$, in the vertical lines of scheme in Fig. 4, the new policy, denoted as $\hat{\boldsymbol{f}}$, is fitted again using the transported $\hat{\boldsymbol{x}}$ and $\dot{\hat{\boldsymbol{x}}}$ labels.

D. Robot Orientation and Stiffness Generalization

However, when learning and controlling the Cartesian robot pose, we must also generalize the desired end-effector orientation.

Let us consider the end effector to be a vector of infinitesimal length with the base $\boldsymbol{x_0}$ on the end effector position and pointing in the direction of the robot orientation during the demonstration, $\boldsymbol{R}_{ee}$. The transportation of the tip of the vector

$$\begin{tikzcd} x \arrow[r, "f"] \arrow[d, "\phi"'] & \dot{x} \arrow[d, "\frac{\partial \phi}{\partial x}"] \\ \hat{x} \arrow[r, "\hat{f}"'] & \dot{\hat{x}} \end{tikzcd}$$

Fig. 4: Mathematical Scheme of Policy Transportation.

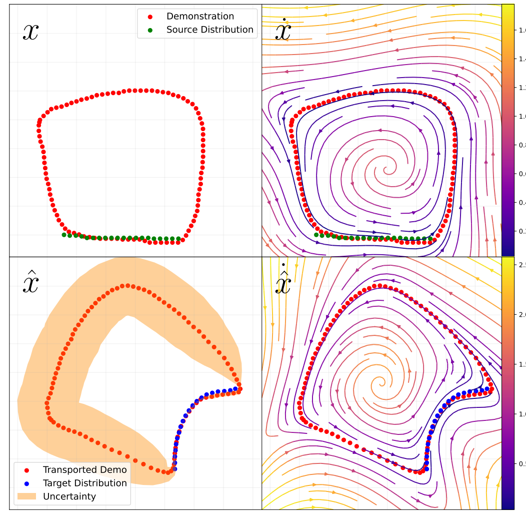

Fig. 5: Graphical representation of the mathematical scheme of policy transportation.

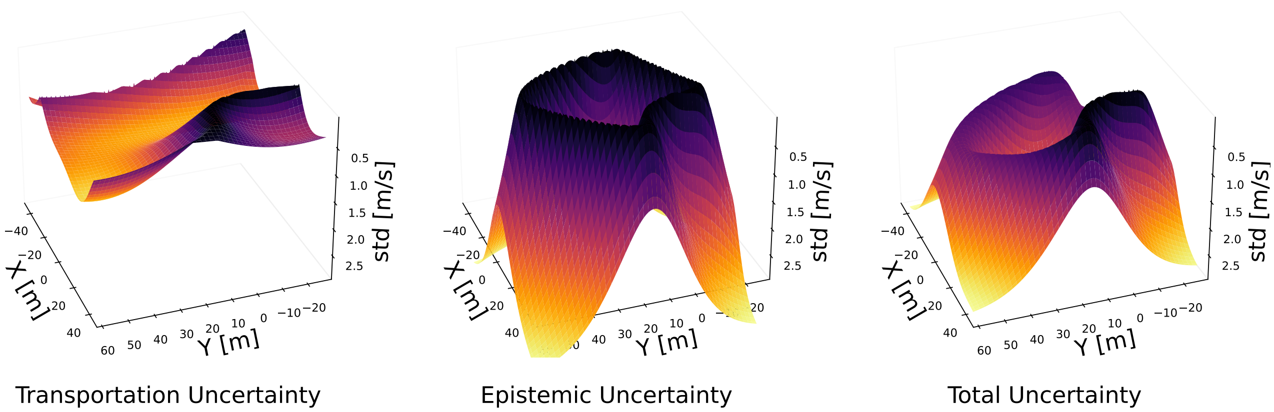

Fig. 6: Standard Deviation quantification on the velocity field. **Transportation Uncertainty** was computed with Eq. (16) and quantifies the (heteroscedastic) uncertainty on the transported label (velocity) corresponding to the transported demonstration. **Epistemic Uncertainty** is the resulting model uncertainty when fitting the new policy $\hat{f}$. **Total Uncertainty** is the resulting standard deviation after computing the variance sum of transportation and epistemic uncertainties.

can be obtained using the Taylor approximation of Eq. (7), according to

$$\hat{\boldsymbol{x}}_{\text{tip}} = \boldsymbol{\phi}(\boldsymbol{x}_{\text{tip}}) \approx \boldsymbol{\phi}(\boldsymbol{x}_{\mathbf{0}}) + \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}}\boldsymbol{\epsilon} = \boldsymbol{\phi}(\boldsymbol{x}_{\mathbf{0}}) + \boldsymbol{J}\boldsymbol{R}_{ee}\boldsymbol{\epsilon}_0 \quad (14)$$

where $\boldsymbol{R}_{ee} \in \mathbb{R}^{3\times 3}$ is the original robot orientation, $\boldsymbol{\epsilon}_0$ is a vector with infinitesimal dimension that has zero orientation. From Eq. (14), it is readily apparent that the transported orientation matrix of the robot end-effector becomes

$$\hat{\boldsymbol{R}}_{ee} := \boldsymbol{J}\boldsymbol{R}_{ee}. \quad (15)$$

The transported orientation matrix needs to be orthogonal with the determinant equal to 1; hence, the pre-multiplication matrix $\boldsymbol{J}$ must have the same properties. We enforce this by normalizing $\boldsymbol{J}$ with its determinant and finding the corresponding orthogonal matrix with a QR decomposition.

Additionally, when implementing policies on a Cartesian impedance control, the stiffness $\boldsymbol{K}_s$ and the damping matrix $\boldsymbol{D}$ must also be transported. The change of coordinates of the stiffness and the damping follows from the transportation of the robot-applied force on the environment found using Hooke's law, i.e.,

$$\hat{\boldsymbol{F}}_s = \hat{\boldsymbol{K}}_s \Delta\hat{\boldsymbol{x}} = \hat{\boldsymbol{K}}_s \overbrace{\boldsymbol{J}\Delta\boldsymbol{x}}^{\Delta\hat{\boldsymbol{x}}} = \boldsymbol{J}\,\overbrace{\boldsymbol{K}_s\Delta\boldsymbol{x}}^{\boldsymbol{F}_s}.$$

Hence, the generalization of the stiffness matrix becomes

$$\hat{\boldsymbol{K}}_s = \boldsymbol{J}\boldsymbol{K}_s\boldsymbol{J}^T$$

and following a similar reasoning for the damping matrix, we obtain,

$$\hat{\boldsymbol{D}} = \boldsymbol{J}\boldsymbol{D}\boldsymbol{J}^T,$$

considering that the inverse of an orthogonal matrix is equal to the transpose of the matrix itself.

E. Transportation Uncertainty

A probabilistic function approximator, like a GP, will also provide the uncertainty on transportation output from Eq. (3) that can be propagated in the transported dynamical system. In particular, a GP derivative is also a GP [26] and its existence will depend on the differentiability of the kernel function. The correlation between derivative samples can be expressed as the second partial derivative $k^{11} = \frac{\partial^2}{\partial x_i \partial x_j} k(x_i, x_j)$ while the correlation between derivative samples and function samples is $k^{10} = \frac{\partial}{\partial x_i} k(x_i, x_j)$. Thus, the mean and variance prediction of the derivative of the Gaussian Process become

$$\begin{aligned} \boldsymbol{\mu'} &= \boldsymbol{K}^{10}_{\boldsymbol{X}_*,\boldsymbol{X}}(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{y} \\ \boldsymbol{\Sigma'} &= \boldsymbol{K}^{11}_{\boldsymbol{X}_*,\boldsymbol{X}_*} - \boldsymbol{K}^{10}_{\boldsymbol{X}_*,\boldsymbol{X}}(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{K}^{01}_{\boldsymbol{X},\boldsymbol{X}_*}. \end{aligned} \tag{16}$$

The uncertainty quantification of the transportation policy becomes essential for calculating the final uncertainty on the control variable, e.g., the velocity. In Fig. 5, the uncertainty is also displayed as a shaded area around the demonstration and as the "warmness" of the color in the vector field.

The uncertainty of the velocity labels is due to the propagation of the original labels through the derivative of the (uncertain) transportation map of (13), i.e.,

$$\boldsymbol{\Sigma}_{\hat{\boldsymbol{x}}} = \boldsymbol{\Sigma}_{\frac{\partial \boldsymbol{\phi}(\boldsymbol{x})}{\partial \boldsymbol{x}}} \dot{\boldsymbol{x}}^{\boldsymbol{2}}. \tag{17}$$

given the definition of the weighted sum of Gaussian variables [30].

Hence, considering that the labels are uncertain, the prediction of the resulting heteroscedastic GP [31] can be computed as the sum of the epistemic and (variable) aleatoric uncertainty, that is

$$\boldsymbol{\Sigma}_{\dot{\hat{x}}} = \boldsymbol{\Sigma}_{\hat{\boldsymbol{f}}} + \boldsymbol{\Sigma}_{\hat{\boldsymbol{x}}}. \tag{18}$$

Fig. 6 depicts the *transportation uncertainty* on the norm of the velocity, calculated with Eq. (16) and the epistemic uncertainty of the model $\hat{\boldsymbol{f}}$, computed with Eq. (3) using transported position and transported velocities labels. From Fig. 5 and 6, it is possible to appreciate that the transportation uncertainties grow when evaluating in regions that are far away from the task parameterization points since the transportation is less certain when going far away from the distribution data; on the other hand, the epistemic uncertainty grows when evaluating in points that are far from the transported demonstration.

In conclusion, the sum of the two uncertainty fields in Fig. 6 grows either when we go far away from the (transported) demonstration or away from the points of the source/target distribution.

V. 2-D Simulations and comparisons

The availability of (calibrated) uncertainties is an important feature to improve the trustworthiness in deploying robot motion generalization. In this section, we evaluate Gaussian Process Transportation (GPT) on generalizing the demonstration in a 2D surface cleaning task and on a reference frame-to-frame motion generalization. Our goal for these simulated experiments is

- to illustrate and compare how the generalization process differs when employing regressors other than a Gaussian Process or methodologies from the state-of-the-art while generalizing a cyclic demonstration that approaches and then retreats from the surface "to clean", in Sec. V-A;
- assess and compare GPT's ability to generalize in multi-reference frame tasks, measuring its performance against state-of-the-art algorithms, in Sec. V-B.

A. 2-D surface cleaning

Fig. 5 visualizes the transportation of the given demonstration, in red, from the source to the target space, using the transportation map $\boldsymbol{\phi}$ where the non-linear component was chosen to be a GP, given the out-of-distribution prediction and the calibrated uncertainty quantification. However, other state-of-the-art function approximators can be used to fit the transportation function without loss of generality. To ensure a fair comparison, the mean linear transformation, i.e., γ, is applied to all trajectories before using the different methods to perform the non-linear transportation. Table I summarises the method with their properties, while Fig. 7 shows the generalization of the demonstration when these methods are used.

Kernelized Movement Primitives (KMP) [6], in this study, fits the motion as a function of time while Laplacian editing (LE) [13] considers the topology of the demonstration to be a chain, i.e., a graph where only consecutive vertices are connected with an edge or as a ring, when the demonstration is periodic, i.e. also starting and ending nodes are connected, like in Fig. 7. Hence, every point of the source distribution is matched with the closest point of the demonstration. Then, each point of the trajectory, or the graph, is moved, knowing the new desired target location of the matched points.

Hence, LE and KMP do not provide any uncertainty on the transportation process.

All the other transportation regressors, a part of the GP, are ensembles of popular regression functions, i.e., Ensemble Random Forest (E-RF). Ensemble Neural Networks (E-NN) and Ensemble Neural Flows (E-NF). An ensemble is a collection of multiple individual models, trained independently, whose combined predictions are used to estimate a distribution on the prediction, i.e., mean and variance. In this example, Neural Networks are simple multi-layer perceptrons while Neural Flows [32] are bijective neural networks, i.e., flows, usually used to learn a mapping from a simple probability distribution to a more complex target distribution.

Fig. 7 depicts the mean and the uncertainty bounds of 2-σ for the transported trajectories when using ensembles and GPs. The bounds are computed from the different fitted models in the ensembles while it is computed analytically for the GP and the depicted GP samples of Fig. 7 are drawn from the posterior distribution.

From Fig. 7, the reader can appreciate how the GP is the only regressor with well-calibrated and unbiased epistemic

TABLE I: Summary table of different methods used to transport trajectories to different surfaces.

Method	Modality	Vel. Gen.	Transportation Uncertainty
KMP [6]	way-points	✗	✗
LE [13]	way-points	✗	✗
E-RF [33]	continuous	✓	estimated
E-NF [24]	continuous	✓	estimated
E-NN [34]	continuous	✓	estimated
GP [26]	continuous	✓	analytical

uncertainty quantification and minimal mean prediction distortion of the trajectory when transporting points far away from the source distribution. For example, the E-NN, has higher uncertainty on the right side of the demonstration, even though the points are at the same distance from the surface, while E-RF generates an un-distorted overconfident transformation, i.e., the uncertainty does not grow when going out of distribution.

B. Multiple Reference Frames

In the literature, one of the main applications of task parameterization is the generalization w.r.t. one or more reference frames. For example, if we teach a robot how to pour water into a glass, we want the robot to automatically generalize the motion w.r.t. any glass position. The task, in this particular case, can be parameterized with the location of the reference frames of each object, which is necessary to track for a successful generalization of the motion. Typically, the motion is projected in any of the reference frames, and a policy is learned w.r.t. each of the frames, leaving out the decision on the relevance of each frame for every timestep. Task-Parametrized Gaussian Mixture Model (TP-GMM) learns a GMM model for the projected demonstration for each of the frames and, during executions, the Gaussians of each frame are combined using the property that the product of Gaussian (PoG) is still a Gaussian, see [2] for more details. Given the GMM, different control formulations are possible, for example only relying on the current state of the system, i.e., $\boldsymbol{\Delta x}_i = \boldsymbol{f}(\boldsymbol{x}_i)$ [35] [2] or by using a Hidden Markov Model (HMM) formulation that also considers the progress during the execution of the trajectory $\boldsymbol{x}_{i+1} = \boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{\alpha}_i)$, where $\boldsymbol{\alpha_i}$, in the context of a mixture mode, selects the properties of the model (mean and variance) that can be used in a tracking algorithm, such as a Linear Quadratic Regulator (LQR) [2], [36]. However, the latent transition matrix between the different states of the HMM is unknown. They need to be estimated using a forward pass algorithm, i.e., the Viterbi algorithm [36], that requires an initial guess trajectory to infer the most likely state transition that generated that initial guess and again generate the most likely motion according to the model. However, having an initial guess can be prohibitive when evaluating the movement in a novel configuration of starting and goal frames.

Differently from these task-parameterized approaches, our proposed method does not track only the reference frame but a set of points that are relevant to the starting and goal object. To guarantee a fair comparison with the state of the art, in Fig. 8, when generalizing using GPT, only 5 points are tracked w.r.t. each reference frame, capturing the position but also the local orientation of the frames. In Fig. 8, what we describe as DMP uses the same mathematical structure of GPT but only relies on a linear transformation, which is why the result is not able to capture the non-linear deformation due to the frame orientation. One of the main perks of our proposed method is the ability to generalize any dynamical system generated by even only one demonstration, unlike GMM-based methods where, to capture a meaningful mixture model, at least two diverse demonstrations need to be provided. Additionally, the GMM uncertainty of the final multi-frame model that results from the PoG does capture the uncertainty of the transportation, while, as depicted in Fig. 8, the GP transportation results in growing uncertainties when transporting points of the demonstrations that are less correlated with the source-target points.

Fig. 8 highlights the discrepancy in the performance of GMM methods on the training set and the test set. At the same time, the reproduction of a known combination of the frames results in accurate rollouts of the policies both when executing them as a dynamical system (TP-GMM)[1] [35] then as an optimal tracking problem of a multi-transition Hidden Markov Model (HMM)[2] [36], when evaluating on the test set, generated on random reorganization of the frames, the resulting trajectories do not successfully reach the goal frame neither in position or orientation. To quantify and compare the different methods, we conducted a quantitative analysis comparing the generalization on known trajectories from the demonstration set or the reaching performance on a randomly generated frame set.

Quantitative Analysis: Fig. 9 shows the box plot that compares the performance of the different models, i.e., TP-GMM, HMM, DMP, and GPT, on the training set. Nine demonstrations are available for different configurations of the starting and goal frame. When training the GMM models, i.e., TP-GMM and HMM, a subset of demonstration m is randomly chosen from the training set and compared with the remaining $(9-m)$ demonstrations when evaluating the model in that unknown situation; the number of used demonstration is highlighted as an apex, e.g., HMM_6 means that we used an HMM model with six demonstrations. When training the transportation models, linear or non-linear, i.e., DMP or GPT, only one demonstration is randomly picked from the training set and compared with the other eight unseen situations. For each model, the random selection of demonstration and comparison is repeated 20 times. Five metrics are used to compare the rollout trajectory and the actual demonstration:

- Frechet distance that does not consider any knowledge of time but finds the maximum distance among all the possible closest pairs among the two curves [37];
- area between the two curves that constructs quadrilaterals between two curves and calculates the area for each quadrilateral [37];

[1]Implementation available at https://github.com/BatyaGG/Task-Parameterized-Gaussian-Mixture-Model

[2]Implementation available at https://gitlab.idiap.ch/rli/pbdlib-python/-/blob/master/notebooks/

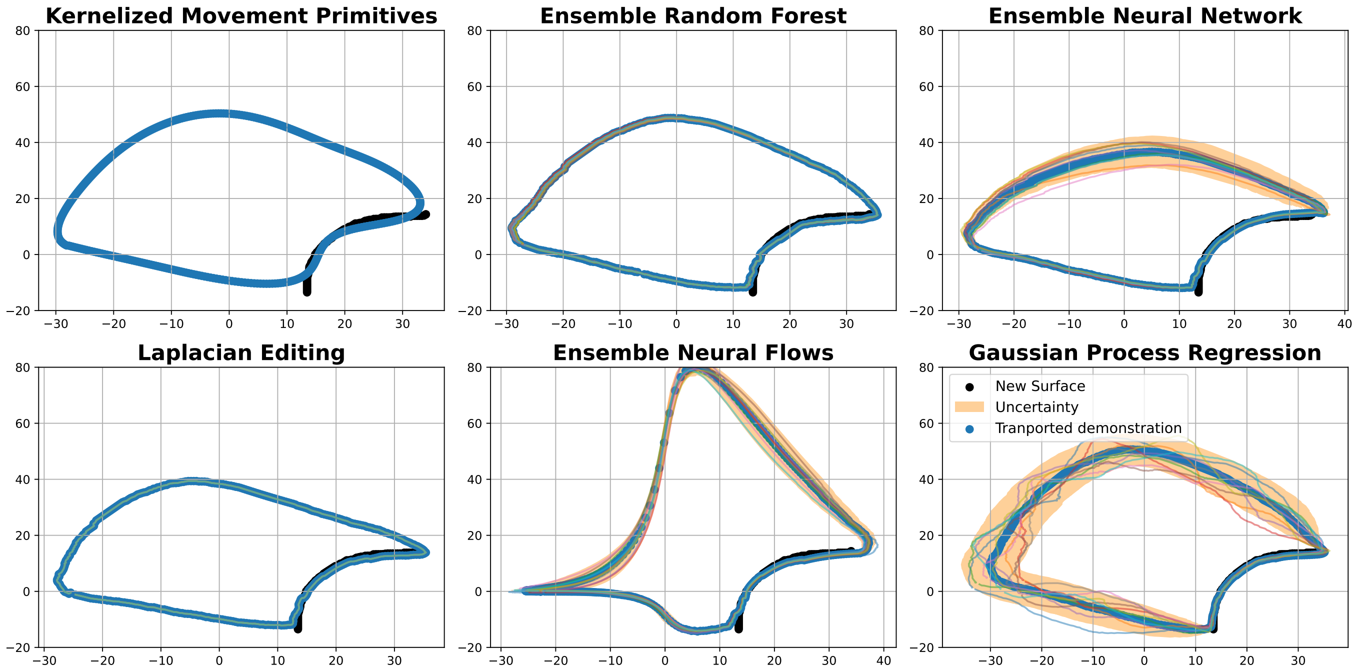

Fig. 7: Qualitative comparison of transportation of demonstration in target space for 2D surface cleaning. The colored lines are the samples of the final transported trajectory policy, i.e. $\hat{x} = \phi(x)$, and the orange area is 2 standard deviation. The black curve is the 1-D surface to clean.

- Dynamic Time Warping (DTW) that computes the cumulative distance between all points in the trajectory [37];
- final position error, computed as the Euclidean distance between the final point of the trajectory and the rollout;
- final trajectory angle, that computes the approach "docking" angle of the trajectory. A low error in the angle distance means that the reproduced trajectory approaches the goal from the same direction as the provided demonstration in the same circumstance.

Considering that we have many models that can behave differently according to the amount and quality of the demonstration, it is not straightforward to deduce any conclusions on which method is statistically better from the boxplot of Fig. 9. For this reason, we run a U test, also known as the Mann-Whitney non-parametric test [38], to deduce if the distribution of results of each of the methods is statistically lower ($p < 0.05$) than each of the others. When computing the U test between two methods, in case of a statistical difference, the winning method gets one point. The numbers on top of the figure for each of the methods indicate the performance ranking, i.e., the method that obtained the most points when computing the U-test is going to be first in the ranking. When more methods share the same position in the ranking, it simply means they were significantly better than the same amount of other methods during the comparisons.

Fig. 9 shows that for Frechet, final position and orientation error, GPT (trained with a single demonstration) performs the best. In contrast, for Area btw the curves and DTW, GPT performs equally or better than GMM and HMM models trained with five demonstrations.

Finally, Fig. 10 shows the box plot and ranking for the model evaluated in a test set with randomly placed frames, and GPT performs statistically better than any other method when reaching the right goal and from the right direction.

C. Multi-source single-target generalization

Fig. 8, in the column of Gaussian Process Transportation, depicts the generalization of a single demonstration from one single 2-frame source to multiple 2-frame targets. Although, in Fig. 1, we already depicted the generalization of many demonstrations and the learned dynamics from one source surface to another target, we still did not mention the generalization from multi-source to a single target. When dealing with n source distributions, we fit n different transportation functions ϕ, each trained with different source points but the same target points.

Fig. 8 highlights how the many demonstrations given in different frame configurations can be transported in the same target frame and how we can extract a reactive policy, encoded in the vector field, as a function of the global position [25].

VI. Real Robot Validation

To validate the proposed method on real manipulation tasks, we selected three challenging tasks, i.e., robot reshelving (Sec. VI-A), dressing (Sec. VI-B) and cleaning (Sec. VI-C), to teach as a single demonstration and generalize it in different scenarios. These are all tasks where the training set will never be similar to the test set; for example, when dressing a human, the configuration and shape of the arm may change, and we expect the robot to generalize the behavior accordingly.

We controlled a Franka Robot using a Cartesian impedance control[3].

The dynamics of the demonstrations are learned with a non-parametric function approximation for motor learning

[3]https://github.com/franzesegiovanni/franka_human_friendly_controllers

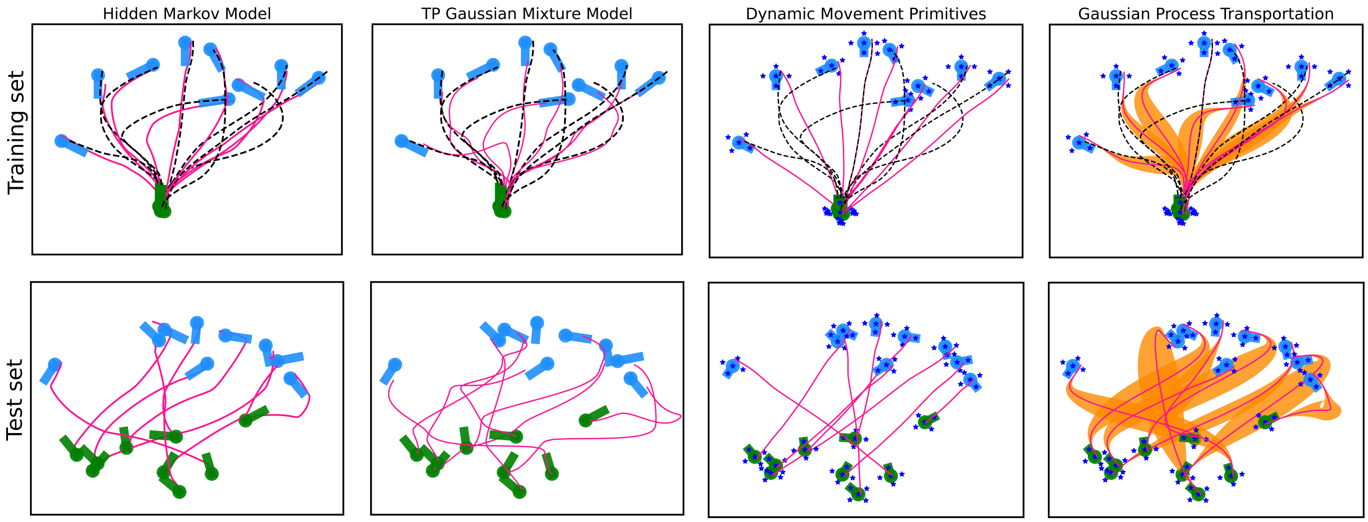

Fig. 8: Qualitative comparison of multi-reference frame parameterization. Comparison between HMM, TP-GMM, DMPs, and our proposed method. The first raw compared the performance in the reproducing of the training set demonstration, depicted as a dashed black line, where both HMM and TP-GMM are training using all the nine demonstrations while DMP and GPT are only trained with one demonstration (the central one) and generalized for each of the frames. In the second row, a random perturbation is applied to each frame, and the model is queried on the most likely trajectory. For GPT, the uncertainty in the generalization is depicted with the orange areas. The blue stars are the points tracked during the motion, given that our proposed method does not rely on reference frames but only on source and target points.

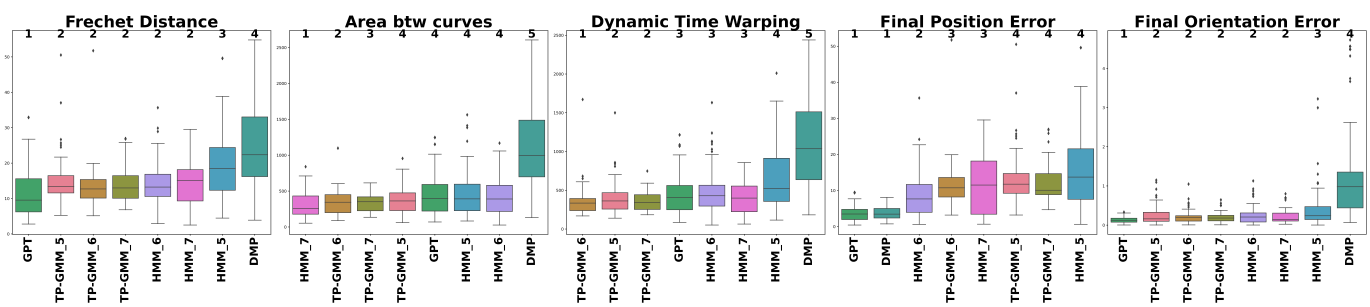

Fig. 9: Box plot results and performance ranking on frame configuration from the training set. The number on top of each box plot is the position of the method in the performance ranking.

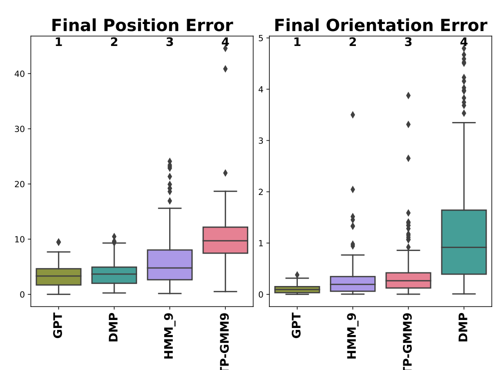

Fig. 10: Box plot results and performance ranking on randomly generated frame configurations.

that uses a joint position-time encoding, proposed in [39]. The distance from the next attractor position and orientation are a function of the current robot position. Our goal is to show how the proposed transportation theory can correctly generalize the pose, velocity, and stiffness of the robot. The following sections will summarize the robot validation experiments. A video of all the experiments can be found at `https://youtu.be/FDmWF7K15KU`.

A. *Robot Reshelving*

Robot reshelving refers to picking an object in one location, moving it, and placing it in a desired position on a shelf.

Our assumptions for the problem are:

- one global frame dynamical system is learned from a single demonstration and transported in the different object/goal configurations;
- corner points of the objects and the shelf slot are tracked rather than position/orientation.

Fig. 12 depicts the experimental setup where a milk box, with an AprilTag [40] on it, has to be positioned on a compartment on a shelf, also marked by another tag. Before the demonstrations or execution, the robot searches for any frames in the spaces using the camera attached to its end-effector. For every frame, the transportation policy extracts

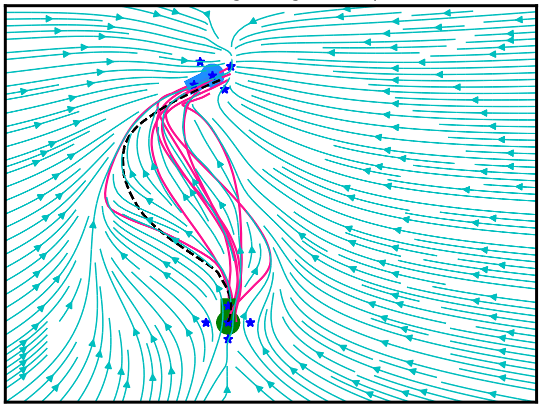

Fig. 11: Multi-source single target generalization. Demonstrations from different frame positions (see Fig. 8 are transported on a single target multi-frame configuration (unknown from the training set). The dashed line is the given human demonstration in that configuration. The vector field is the resulting dynamics learned also with a Gaussian Process with minimization of uncertainties from [25].

TABLE II: Range of Variability for Object and Goal Frames.

Frame	x [m]	y [m]	z [m]	yaw [deg]
Object	0.225	0.366	-	94.6
Goal	0.337	0.036	0.675	-

a cube's center and corners with predefined side dimensions as the markers. Fig. 12 shows how the demonstration for reshelving on the left of the central compartment can be generalized to any other floor, both on the left and right. We randomized the object position and orientation and the goal on the shelf ten different times, all successfully generalized. Table II shows the range of x,y,z, and yaw angles of the object and goal markers during the ten different executions, while Fig. 13 depicts the relative position w.r.t. the object and the frame of the different rollouts; from the figure it is possible to appreciate how the execution lines converge on the (initial) object position when picking and on the goal position when placing the object.

B. Robot Dressing

The task of dressing is a primary task in elderly care. It consists of pulling a deformable sleeve over the posture of a human arm.

Complicated motions need to be executed by the robot to increase the dressing success rate, i.e., reaching the shoulder without getting stuck or exercising too large force on the arm. Fig. 14 depicts the robot experimental setup where an articulated mannequin is posed in different shoulder positions and arm configurations. Four AprilTags [40] are glued on the arm, shoulder, elbow, wrist, and hand, captured by the camera on the robot wrist at the beginning of each demonstration/execution. From the markers, only the position is extracted this time. The piece of cloth is pinched in the end effector by the user before starting the experiments. We leverage the assumption that the pose will not change during the demonstration; however, it is worth mentioning that the arm structure is not fixed on the table, so if the generalization is not good and the robot maliciously touches the arm, the resulting displacement would result in unsuccessful dressing. Only one demonstration was given to the robot. Then, the arm was reset for a different range of x,y positions of the shoulder and configuration of the arm. The ranges of variation of the task parameters are $\Delta x_{\text{shoulder}} = 0.122$ [m], $\Delta y_{\text{shoulder}} = 0.259$ [m], $\Delta\alpha = 56$ [deg], where α is the angle that the elbow intercepts with the connecting line between the shoulder and the wrist. A fully stretched arm (easy pose to dress) has $\alpha = 180$, and when the hand touches the shoulder (impossible pose to dress) $\alpha = 0$. The policy transportation was able to generalize the policy for every requested arm configuration.

C. Robot Surface Cleaning

Surface cleaning/grinding tasks require robots to not only track surface shapes but also apply the right amount of force for successful cleaning/grinding. Robotic cleaning or grinding involves automated machines equipped with specialized tools to perform cleaning tasks.

In this experiment, we want to show that

- we can learn a general policy that may involve polishing phases and free movement phases;
- we do not need any force sensors to align to the surface;
- the surface is unknown, and only an ordered point cloud is obtained from the camera sensors.

One of the main advantages of the proposed method is that it does not need to reconstruct the surface but only learns the map from the source to the target pointcloud. The deformation between the source and the target surface point cloud is modeled using a Stochastic Variational Gaussian Process Transportation (SV-GPT) to generalize the demonstrated policy position, orientation, and stiffness profile for a successful cleaning task. Given the large number of points in the source and target point cloud, i.e., 400 points, using a reduced set of inducing points, i.e., 100, makes fitting the transportation model more computationally efficient.

Fig. 15 depicts the teaching of a cleaning task on a flat surface and the generalization on different higher, titled, and curved surfaces that belong to common objects. The lower row shows what the robot perceives of the environment; the blue dots in space are the source distribution, recorded before giving the demonstration (depicted as dashed line), and target distributions recorded before executing the roll-out transported policy (depicted a solid line). Fig. 15 also highlights how the roll-outs follow the shape of the surface, showing a successful generalization of the robot position and orientation. As previously stated, no external force-torque sensor is used to adapt the orientation of the end-effector on the tangential direction of the surface. However, an observer of the applied external force between the robot and the surface is estimated from measured torques in the joints. Fig. 16, depicts the estimated norm of the force exchanged with the surface, where the same increasing/decreasing trend is captured on the different surfaces.

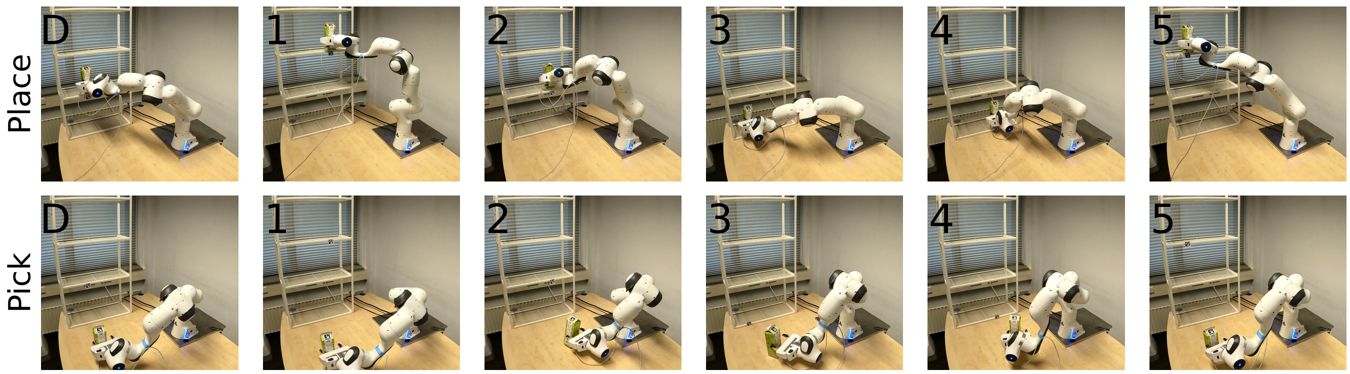

Fig. 12: Generalization of the reshelving task. The first column is the robot reproduction in the demonstration scenario.

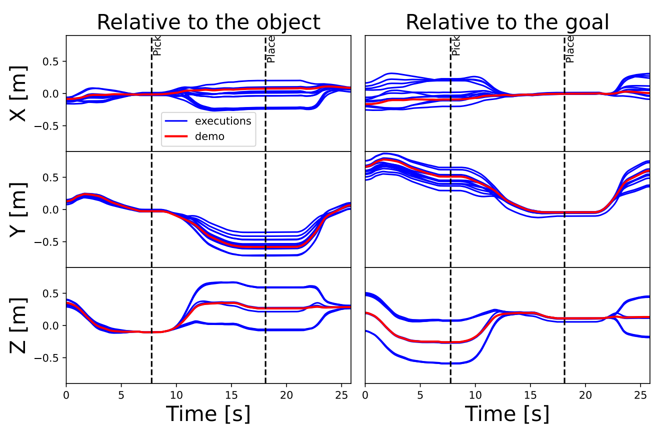

Fig. 13: Relative position of the end-effector w.r.t. the initial object and the goal position during multiple generalizations rollouts in robot reshelving.

VII. Limitations and Open Challenges

Despite the successful application of the proposed policy transportation on different challenging tasks, we can foresee some limitations and future challenges to improve the applicability and have a broader impact.

For example, we assume knowing the matching between the points of the source and the target distribution. However, in many complex scenarios, this limitation can be problematic, and some different pre-processing algorithms, such as optimal transport [41] or iterative closest point (ICP), need to be adopted to perform the matching. Additionally, semantic matching can increase cross-domain generalization, for example, by adapting the reshelving strategy to a completely different shelf type or adjusting the dressing policy from an adult to a baby arm.

Another assumption of the developed method is dealing with static environments, i.e., the target distribution does not change during the policy's rollout. However, this assumption can fall when dealing with the reshelving of moving objects or when trying to dress real humans that will probably move before and during the interaction. Nevertheless, supposing to know the state of the target distribution, the transportation policy can be updated inexpensively online since only the label $\hat{y}$ of Eq. (7) and the (cheap) linear transformation needs to be recomputed. However, the fitting of the transported policy $\hat{f}$ makes it challenging to perform the generalization online.

Finally, given that in complex scenarios, the generalization may be inaccurate, the use of interactive human corrections may increase the resulting manipulation performance [1]. However, changing the generalized policy opens the question of whether interactive corrections should be propagated back to the source policy and how. Additionally, in case many source distributions/policies are recorded, the choice of generating the target policy by transporting all of them, like in Fig. 8, or by selecting the best one, according to some similarity criteria, can open exciting developments of the proposed theory.

VIII. Conclusions

In this paper, we address the prominent but challenging problem of policy generalization to novel unseen task scenarios. We formulate a novel policy transportation theory that, given a set of matched source and target points in the task space of the robot, regresses the function that, most likely, would match the source and target distribution. Additionally, we showed how the same transportation function and its derivatives can be exploited to transport the original policy dynamics, rotation, and stiffness while retaining uncertainties in the process. The same algorithm, which uses a Gaussian Process at its core, was tested and compared with different state-of-the-art regressors or different generalization methods, showing how, even with only one source demonstration, it results in better or comparable performance. However, the main requirement, for a successful generalization is to track and match important task points in the original scenario, where the demonstration was given, and the corresponding points in the new scenarios.

We validated the proposed approach on a Franka Robot, testing it on three different tasks: product reshelving, arm dressing, and surface cleaning. These various tasks were never tackled together by the same generalization algorithm, and they usually were performed with ad hoc solutions, for example, to keep a constant force when cleaning a surface.

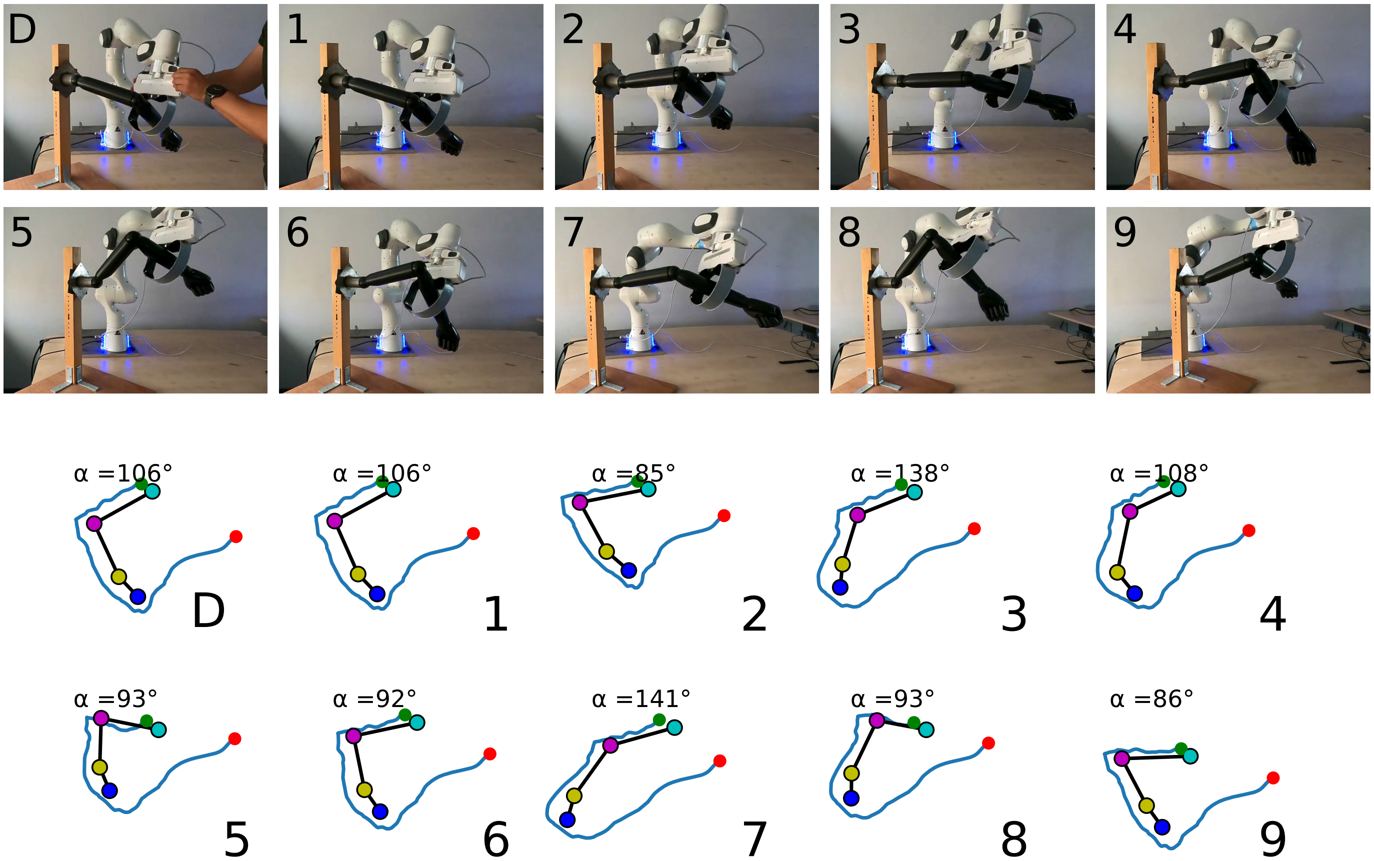

Fig. 14: Dressing policy generalization. Cyan marker is the shoulder, magenta is the elbow, yellow is the wrist, and blue is the hand. The blue rollout end-effector trajectory starts from the red dot and finishes with the green dot. α is the angle of the elbow; the smaller the angles, the more complicated the generalization would be.

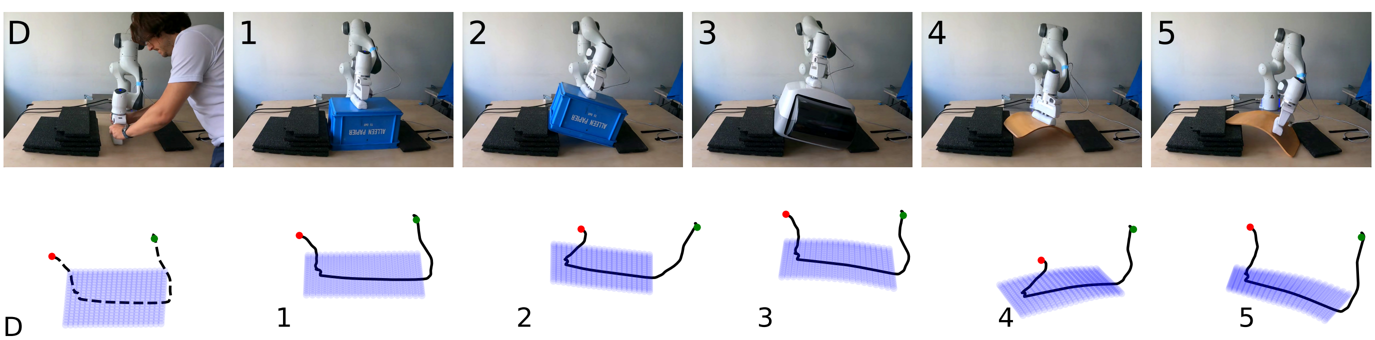

Fig. 15: Point cloud, demonstration and rollouts of the generalized motion in cleaning tasks.

Despite this, our policy transportation algorithm performed successfully in all of them. The tracking requirements were satisfied using fiducial markers or directly the point cloud estimated with the infrared camera sensor. Future development will have to focus on scaling the process on big and unmatched point clouds of complex (and deformable) objects to manipulate while allowing the use of human feedback in the fine-tuning of the resulting policy.

References

[1] C. Celemin, R. Pérez-Dattari, E. Chisari, G. Franzese, L. de Souza Rosa, R. Prakash, Z. Ajanović, M. Ferraz, A. Valada, and J. Kober, "Interactive imitation learning in robotics: A survey," *Foundations and Trends® in Robotics*, vol. 10, no. 1-2, pp. 1–197, 2022.

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

[3] M. Saveriano, F. J. Abu-Dakka, A. Kramberger, and L. Peternel, "Dynamic movement primitives in robotics: A tutorial survey," *The International Journal of Robotics Research*, vol. 42, no. 13, pp. 1133–1184, 2023.

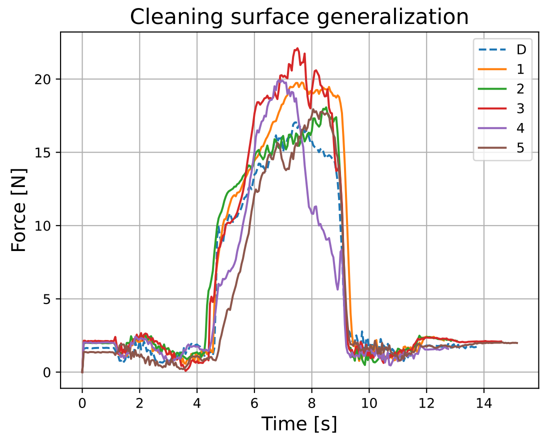

Fig. 16: Norm of the force perceived [N] from the end-effector when executing the transported dynamics on the new surfaces.

[4] M. Saveriano, F. Franzel, and D. Lee, "Merging position and orientation motion primitives," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 7041–7047.

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

[6] 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.

[7] S. Calinon, F. Guenter, and A. Billard, "On learning the statistical representation of a task and generalizing it to various contexts," in *2006 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2006, pp. 2978–2983.

[8] ——, "On learning, representing, and generalizing a task in a humanoid robot," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 2, pp. 286–298, 2007.

[9] S. Calinon, F. D'halluin, D. G. Caldwell, and A. G. Billard, "Handling of multiple constraints and motion alternatives in a robot programming by demonstration framework," in *2009 9th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2009, pp. 582–588.

[10] S. Calinon, Z. Li, T. Alizadeh, N. G. Tsagarakis, and D. G. Caldwell, "Statistical dynamical systems for skills acquisition in humanoids," in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. IEEE, 2012, pp. 323–329.

[11] Y. Huang, J. Silvério, L. Rozo, and D. G. Caldwell, "Generalized task-parameterized skill learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 5667–5474.

[12] A. Sena, B. Michael, and M. Howard, "Improving task-parameterised movement learning generalisation with frame-weighted trajectory generation," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4281–4287.

[13] T. Nierhoff, S. Hirche, and Y. Nakamura, "Spatial adaption of robot trajectories based on Laplacian trajectory editing," *Autonomous Robots*, vol. 40, pp. 159–173, 2016.

[14] T. Li and N. Figueroa, "Task transfer with stability guarantees via elastic dynamical system motion policies," in *7th Annual Conference on Robot Learning*, 2023.

[15] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel, "Laplacian surface editing," in *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, 2004, pp. 175–184.

[16] A. Mészáros, G. Franzese, and J. Kober, "Learning to pick at non-zero-velocity from interactive demonstrations," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6052–6059, 2022.

[17] R. P. Joshi, N. Koganti, and T. Shibata, "A framework for robotic clothing assistance by imitation learning," *Advanced Robotics*, vol. 33, no. 22, pp. 1156–1174, 2019.

[18] E. Pignat and S. Calinon, "Learning adaptive dressing assistance from human demonstration," *Robotics and Autonomous Systems*, vol. 93, pp. 61–75, 2017.

[19] J. Zhu, M. Gienger, and J. Kober, "Learning task-parameterized skills from few demonstrations," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4063–4070, 2022.

[20] W. Amanhoud, M. Khoramshahi, M. Bonnesoeur, and A. Billard, "Force adaptation in contact tasks with dynamical systems," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6841–6847.

[21] L. Armesto, J. Moura, V. Ivan, M. S. Erden, A. Sala, and S. Vijayakumar, "Constraint-aware learning of policies by demonstration," *The International Journal of Robotics Research*, vol. 37, no. 13-14, pp. 1673–1689, 2018.

[22] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with Gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[23] R. Pérez-Dattari and J. Kober, "Stable motion primitives via imitation and contrastive learning," *IEEE Transactions on Robotics*, vol. 39, no. 5, pp. 3909–3928, 2023.

[24] M. A. Rana, A. Li, D. Fox, B. Boots, F. Ramos, and N. Ratliff, "Euclideanizing flows: Diffeomorphic reduction for learning stable dynamical systems," in *Learning for Dynamics and Control*. PMLR, 2020, pp. 630–639.

[25] G. Franzese, A. Mészáros, L. Peternel, and J. Kober, "ILoSA: Interactive learning of stiffness and attractors," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 7778–7785.

[26] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[27] M. Titsias, "Variational learning of inducing variables in sparse Gaussian processes," in *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, vol. 5. PMLR, 2009, pp. 567–574.

[28] N. Courty, R. Flamary, A. Habrard, and A. Rakotomamonjy, "Joint distribution optimal transportation for domain adaptation," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[29] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 5, pp. 698–700, 1987.

[30] M. P. Deisenroth, A. A. Faisal, and C. S. Ong, *Mathematics for machine learning*. Cambridge University Press, 2020.

[31] Q. V. Le, A. J. Smola, and S. Canu, "Heteroscedastic Gaussian process regression," in *22nd International Conference on Machine Learning*, 2005, pp. 489–496.

[32] I. Kobyzev, S. J. Prince, and M. A. Brubaker, "Normalizing flows: An introduction and review of current methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 11, pp. 3964–3979, 2020.

[33] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[34] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[35] T. Alizadeh and B. Saduanov, "Robot programming by demonstration of multiple tasks within a common environment," in *2017 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE, 2017, pp. 608–613.

[36] S. Calinon, A. Pistillo, and D. G. Caldwell, "Encoding the time and space constraints of a task in explicit-duration hidden Markov model," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 3413–3418.

[37] C. F. Jekel, G. Venter, M. P. Venter, N. Stander, and R. T. Haftka, "Similarity measures for identifying material parameters from hysteresis loops using inverse analysis," *International Journal of Material Forming*, vol. 12, pp. 355–378, 2019.

[38] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The Annals of Mathematical Statistics*, pp. 50–60, 1947.

[39] G. Franzese, L. de Souza Rosa, T. Verburg, L. Peternel, and J. Kober, "Interactive imitation learning of bimanual movement primitives," *IEEE/ASME Transactions on Mechatronics*, pp. 1–13, 2023.

[40] J. Wang and E. Olson, "AprilTag 2: Efficient and robust fiducial detection," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4193–4198.

[41] Y. Luo, Z. Jiang, S. Cohen, E. Grefenstette, and M. P. Deisenroth, "Optimal transport for offline imitation learning," in *The Eleventh International Conference on Learning Representations*, 2023.