TU Delft

Delft University of Technology

Generalizable Motion Policies Through Keypoint Parameterization and Transportation Maps

Franzese, Giovanni; Prakash, Ravi; Santina, Cosimo Della; Kober, Jens

DOI
10.1109/TRO.2025.3582821

Publication date
2025

Document Version
Final published version

Published in
IEEE Transactions on Robotics

Citation (APA)
Franzese, G., Prakash, R., Santina, C. D., & Kober, J. (2025). Generalizable Motion Policies Through Keypoint Parameterization and Transportation Maps. *IEEE Transactions on Robotics*, *41*, 4557-4573. https://doi.org/10.1109/TRO.2025.3582821

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Green Open Access added to TU Delft Institutional Repository
as part of the Taverne amendment.

More information about this copyright law amendment
can be found at https://www.openaccess.nl.

Otherwise as indicated in the copyright section:
the publisher is the copyright holder of this work and the
author uses the Dutch legislation to make this work public.

Generalizable Motion Policies Through Keypoint Parameterization and Transportation Maps

Giovanni Franzese, Ravi Prakash, *Member, IEEE*, Cosimo Della Santina, *Senior Member, IEEE*, and Jens Kober, *Senior Member, IEEE*

***Abstract*—Learning from Interactive Demonstrations has revolutionized the way nonexpert 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 keypoints is tracked during the demonstration and the execution, e.g., a point cloud of the surface to clean. We then propose to fit a nonlinear 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 task keypoints 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.**

***Index Terms*—Imitation learning, one shot learning, robot learning, statistical learning.**

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.

Received 28 October 2024; revised 4 May 2025; accepted 14 May 2025. Date of publication 24 June 2025; date of current version 29 July 2025. This work was supported in part by the National Growth Fund program NXTGEN Hightech and by the European Research Council Starting Grant TERI Teaching Robots Interactively (Project 804907). This article was recommended for publication by Associate Editor M. Walter and Editor J. Bohg upon evaluation of the reviewers' comments. *(Corresponding author: Giovanni Franzese.)*

Giovanni Franzese, Cosimo Della Santina, and Jens Kober are with the Cognitive Robotics, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: g.franzese@tudelft.nl; c.dellasantina@tudelft.nl; j.kober@tudelft.nl).

Ravi Prakash is with the Cyber Physical Systems, Indian Institute of Science Bangalore, Bangalore 560012, India (e-mail: ravipr@iisc.ac.in).

A video of the experiments can be found here: https://youtu.be/bE6uOnAQBLo.

This article has supplementary downloadable material available at https://doi.org/10.1109/TRO.2025.3582821, provided by the authors.

Digital Object Identifier 10.1109/TRO.2025.3582821

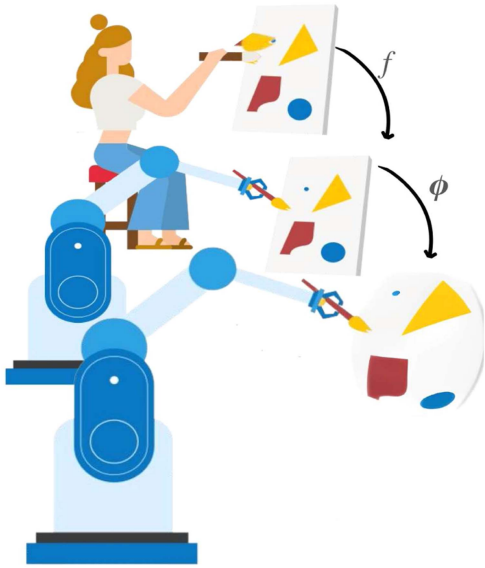


Fig. 1. Example of Policy Transportation. The human demonstrates to the robot how to perform a task on a flat canvas. Then, the robot, when facing a new curved canvas, "transports" its knowledge in the new situation by adapting the end effector velocity, orientation, and stiffness to correctly adapt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

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. A versatile task parametrization to describe different situations, such as object pose, arm posture, and surface shape, is through a set of keypoints [3].

However, the current state of the art in task parameterization struggles to scale when managing numerous task parameters, such as describing a surface as a set of points, and fails to generalize effectively for policy states that are not in close proximity to the parameterization points due to the covariate shift.

We propose the learning of a "transportation" map that is trained to align a set of keypoints from the original task configuration (e.g., a flat surface) to a new configuration (e.g., a curved surface) as illustrated in Fig. 1. This map is then used

1941-0468 © 2025 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

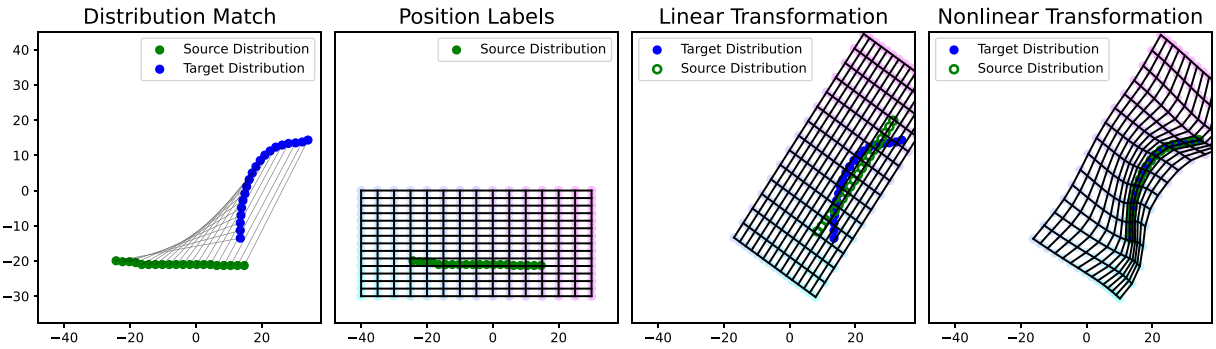


Fig. 2. 2-D transportation. **Distribution match** depicts the source and the target distribution correspondence used to train the transportation function. **Source Distribution** depicts a grid of points in the original space. **Linear Transformation** shows the effect on the original grid when only an affine transformation γ is used to match source and target distribution. **Nonlinear Transportation** captures the deformation of the space when the source points are forced to match the target ones via the application of the complete policy ϕ.

to generalize the labels of the original policy, such as position, velocity, stiffness, and orientation. Given that the original policy's set of labels might have been interactively adjusted or aggregated based on human feedback [1], [4], transforming the demonstration alone is insufficient. This limitation makes trajectory reshaping methods inappropriate for achieving the generalization goal [5]. Therefore, we propose a method that not only works for the generalization of demonstrations encoded as trajectories but also if they are encoded as a set of *independent* labels. The proposed algorithm will "transport" the original labels on the new task configuration where a new policy can be re-fitted, making the generalization algorithm agnostic to the method used for policy learning, e.g., DMP [6], KMP [7], GP [4], NN [8], GMM [9], LPV [10], etc.

Moreover, our goal is to generalize out-of-distribution policies, e.g., motions that are far away from the table to clean. For this reason, it is important to define a transportation function that has desired out-of-distribution (o.o.d.) properties and is able to correctly quantify the uncertainties of the generalization. Ideally, the robot would generalize the learned skill to novel o.o.d. situations without requiring the aggregation of hundreds of thousands of demonstrations [11], regulate its stiffness as a function of the uncertainty [12], or actively ask for data in uncertain situations [4].

The next sections will highlight how this article contributes to the field of robot policy generalization with the formalization and testing of a policy transportation theory, Sections III-A–III-D that can do the following:

1) transport the demonstration position labels from the original task space to the new (deformed) space, see Section III-E;
2) transport the position labels, velocity labels, end-effector orientations, stiffness, and damping by exploiting the derivative of the transportation mapping, see Section III-F;
3) estimate the final uncertainty due to the reduced set of demonstrations and the estimated uncertainty in the transportation map, see Section III-H.

The proposed 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 Section V.

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) [6] 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 was addressed in [13], 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 and generalize the task with respect to the goal [14]. Approaches for modeling and generalizing demonstrations that have shown improved performances with respect to the DMPs are Probabilistic Movement Primitives (ProMPs) [15]. 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) [7] proposed a nonparametric 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

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

the original trajectory or rely on a heuristic that, for example, matches each waypoint with the closest point in the trajectory.

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 (LE) [5], [16]. It uses the Laplace-Beltrami operator, a well-known algorithm in the computer graphics community, to deform meshes [17], 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. Similarly, [18] proposed a conditioning operation to specify the desired waypoint at a specific time of the motion. However, this approach is very specific for trajectory reshaping or movements that are encoded as a function of time and requires explicit knowledge of the new via point for some trajectory nodes or time instances.

Gaussian mixture models (GMM) [19], [20], [21] 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], [22], [23], [24]. The local models are then fused in global coordinates, using the Product of Gaussians (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. Moreover, [25] also uses a Gaussian Mixture Regression to adopt a linear limit cycle to a nonlinear one by learning a modification of the motion amplitude as a function of the phase. However, this is very specific to phase-dependent motions, where the learned phase-dependent map would never have to be evaluated for out-of-distribution phases.

A different task-parameterized approach to generalization as formulated in [26] uses Task-Parameterized Equation Learner Networks that have as features the task parameters and the time of each demonstration and given a set of demonstrations it can regress the equations that describe the desired position as a function of time and desired task parameters, such as the desire hight of the goal. By avoiding overfitting, the method also performs well on task parameters that are outside the boundary of the observed range. However, since the parameters are considered as extra features of the regression, *very far* extrapolations are not possible; moreover, the method requires multiple time-dependent demonstrations to regress the equations. We aim to relax both of these requirements and to extrapolate to extreme task parameter changes.

In this article, we rely on a convenient 3-D keypoint parametrization since the points could be extracted from a tracked object/reference frame or a point cloud. Moreover, we relax the need to explicitly specify the new via points for a specific node in the trajectory since the via point definition, i.e., which point of the trajectory should be moved, can be error-prone and require ad-hoc assignment algorithms between the keypoints and the trajectory. Yet, using keypoints is a double-edged sword. Although keypoints are a more expressive description of the task [3], if they are not moving rigidly, we cannot simply roto-translate our policy [27], [28]. We would have to rely on some planning [29] or sim-2-real RL [30]. We are the first to contribute a nonlinear transformation in this context of imitation learning to modify the original labels and show successful zero-shot motion generalization to different surface shapes, different arm configurations in dressing, and different object locations and goal locations.

The requirement of having a bijective map is usually used to enforce the asymptotic stability of nonlinear policy [31] [32] or to couple the motion of two manipulators to perform bilateral teleoperation [33]. However, in this article, we relax the requirement on the invertibility of the map, similarly to [34], and moreover, we use the map to transfer dynamics, orientation, and stiffness, and not only to couple the motion of two manipulators [33].

Robot Reshelving: The authors of [35] propose, within the realm of robotic retail automation, to enable nonexpert 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 systems have been applied to learn dressing tasks; for example, 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 [36], TP-HMM [37] and a TP-GMM [38].

Robot Surface Cleaning: Efficient and fast generalization of robotic surface cleaning has been achieved using task-parameterized learning. In [39], 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 nonlinear 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, [40] generalizes the polishing task on the novel curved surface by adapting the orientation and the direction of the contact force to minimize the perceived torque.

The following section formalizes the concept of policy generalization using transportation maps.

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

III. Policy Transportation

A. Problem Setting

When learning a robot policy from (interactive) demonstration, a set of desired state-action pairs is aggregated. Throughout the article, we denote the available *policy labels* as follows.

1) The set of M Cartesian position labels $\boldsymbol{\mathcal{X}} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_M\}$, where $\boldsymbol{x}_i \in \mathbb{R}^3$.
2) The set of M Cartesian velocity labels $\dot{\boldsymbol{\mathcal{X}}} = \{\dot{\boldsymbol{x}}_1, \dot{\boldsymbol{x}}_2, \ldots, \dot{\boldsymbol{x}}_M\}$, where $\dot{\boldsymbol{x}}_i \in \mathbb{R}^3$.
3) The set of M Cartesian orientation labels $\boldsymbol{\mathcal{R}} = \{\boldsymbol{R}_1, \boldsymbol{R}_2, \ldots, \boldsymbol{R}_M\}$, where $\boldsymbol{R}_i \in SO(3)$.
4) The set of M Cartesian stiffness labels $\boldsymbol{\mathcal{K}} = \{\boldsymbol{K}_1, \boldsymbol{K}_2, \ldots, \boldsymbol{K}_M\}$, where $\boldsymbol{K}_i \in S_3^+$.
5) The set of M Cartesian damping labels $\boldsymbol{\mathcal{D}} = \{\boldsymbol{D}_1, \boldsymbol{D}_2, \ldots, \boldsymbol{D}_M\}$, where $\boldsymbol{D}_i \in S_3^+$.

where $\mathbb{R}^3$ denotes the 3-D Euclidean space, $SO(3)$ represents the special orthogonal group of 3×3 rotation matrices, and S_3^+ is the manifold of 3×3 symmetric positive semidefinite matrices.

To develop a task parameterization that scales from pick-and-place to continuous surfaces, we track a set of environment-specific *keypoints* that describe the situation. Our task parameterization comprises N tracked points in $\mathbb{R}^3$, recorded in the demonstration scenario as the source distribution, while the corresponding moved points in the new scenario are defined as the target distribution, defined as the following two *ordered* sets.

1) $\boldsymbol{\mathcal{S}} = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_N\}$, where $\boldsymbol{s}_i \in \mathbb{R}^3$.
2) $\boldsymbol{\mathcal{T}} = \{\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_N\}$, where $\boldsymbol{t}_i \in \mathbb{R}^3$.

Here, we ask these sets to be ordered as we assume that the points of the target and source distributions are already paired. This is a reasonable assumption as this pairing can either be done manually or autonomously via state-of-the-art solutions [41], [42].

Goal: To summarize, our goal is to find a way of learning a policy in the form of a dynamical system in the target space $\dot{\hat{x}} = g(\hat{x})$ by integrating the knowledge of the labels provided in the original space $(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})$ with the knowledge of the keypoints $(\boldsymbol{\mathcal{S}}, \boldsymbol{\mathcal{T}})$, giving us information on how original and target space are related. As a secondary goal, we wish to make use of the labels $\boldsymbol{\mathcal{R}}, \boldsymbol{\mathcal{K}}, \boldsymbol{\mathcal{D}}$ in the novel task, and possibly also quantify the uncertainties of the generalization process.

B. Proposed Solution: Policy Transportation

We propose to solve this challenge via a two step process (see Fig. 3) as follows.

1) First, we learn a policy transportation, mapping the labels $(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}})$ in the target space $(\hat{\boldsymbol{\mathcal{X}}}, \dot{\hat{\boldsymbol{\mathcal{X}}}})$ in a way coherent with the keypoints.
2) Second, we use state-of-the-art techniques to learn the dynamics g from $(\hat{\boldsymbol{\mathcal{X}}}, \dot{\hat{\boldsymbol{\mathcal{X}}}})$.

The rest of the section will, therefore, focus on defining, analyzing, and learning the transportation map of Step 1. We start here by introducing the general definition of the policy transportation. If the number of keypoints is larger than the size of the embedding space (3 in this work), it is, in general, not possible to match them with a linear transformation. Therefore, our policy transportation will necessarily need to be nonlinear. Let us define it as a continuously differentiable (C^1) map

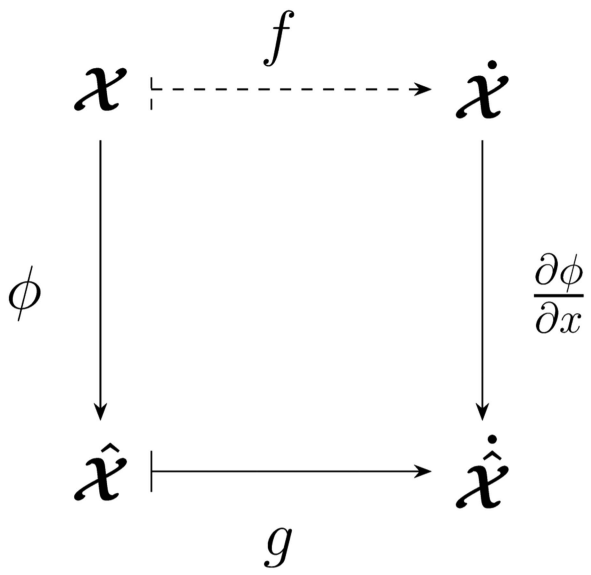


Fig. 3. Mathematical scheme of policy transportation.

$$\boldsymbol{\phi} : \mathbb{R}^3 \to \mathbb{R}^3. \tag{1}$$

Note that our secondary goal can also be achieved as a beneficial by-product of the proposed solutions. We will discuss later in this section how all the remaining labels will be transformed and the uncertainty quantified.

C. Desirable Properties for ϕ

Desired properties: The following are desirable properties ϕ of(1) to fulfill:

1) all keypoints are properly matched—i.e., $\boldsymbol{\mathcal{T}} = \boldsymbol{\phi}(\boldsymbol{\mathcal{S}})$;
2) the policy transportation is a change of coordinates—i.e., $\det(J_\phi) \neq 0$ everywhere.

The reason behind the first property should be clear, as it directly reflects the problem statement.

The reason for considering the second property is that, in general, we would like ϕ to transform the labels in such a way it conserves stability properties. This is not necessary to our method, but it would be beneficial since conserving stability properties may result in better performance. Assume that $(\dot{\boldsymbol{\mathcal{X}}}, \boldsymbol{\mathcal{X}})$ are samples extracted from an ideal underlying dynamical system $\dot{x} = f(x)$. Then, we would like the policy g to have the same stability properties as f. For example, if f is a contractive field, then the dynamics in the new coordinates should also be contractive. A necessary and sufficient condition for this is that 2) is fulfilled [43], [44], [45].

It is important to stress that of the two properties, 1) is definitely the most important, as it ensures that geometrically, the two spaces are properly matched. Instead, note that even if 2) is not fulfilled, this will not jeopardize the stability of the learned policy. After transportation, data can be fit with a stable dynamical system, no matter the choice of ϕ. Actually,

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

the fact that we do not need to compute J_ϕ^{-1} is a key benefit of our method. As such, we will impose property 1) as part of the learning process, while property 2) will only be checked experimentally a posteriori.

D. Convenient Choice for the Transportation Map

Taking inspiration from the definition of registration maps in the point cloud registration literature [42], we reformulate $\boldsymbol{\phi}$ as follows:

$$\boldsymbol{\phi}(\boldsymbol{x}) := \boldsymbol{\gamma}(\boldsymbol{x}) + \boldsymbol{\psi}(\boldsymbol{\gamma}(\boldsymbol{x})). \tag{2}$$

Here, $\boldsymbol{\gamma}$ is an affine transformation and $\boldsymbol{\psi}$ a residual nonlinear transformation. This way, we ensure that the nonlinear transformation $\boldsymbol{\psi}$ only addresses the residual mismatch error remaining after the affine transformation $\boldsymbol{\gamma}$. The affine transformation will, therefore, act globally, and the nonlinearities will only locally add extra deformations to fine-tune the matching process. Fig. 2 shows an example of such a policy transformation in action.

Before diving into the proposed two-step process for learning $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$, such that they fulfill property 1), we delve into some basic analytical considerations on conditions under which property 2) is verified.

Proposition: If $\boldsymbol{\phi}$ fulfills property (i), then a necessary condition for it to also fulfill property (ii) is $\det(J_\phi(\mathcal{S})) > 0$ or $\det(J_\phi(\mathcal{S})) < 0$, where the inequality is to be intended element-wise. The condition becomes also sufficient if we ask it to be true across all possible choices of $\mathcal{S}$.

Proof: We wish to prove that $\det(J_\phi) \neq 0$. The sufficient part is trivially implied by the hypothesis, as by taking all possible choices of $\mathcal{S}$, we are sampling all the support of J_ϕ.

For the necessary part, let us assume that the hypothesis is violated, i.e. there exist two elements $s_i, s_j \in \mathcal{S}$ such that $\det(J_\phi(s_i)) < 0$ and $\det(J_\phi(s_j)) > 0$. We define a generic continuous path $p : [0, 1] \to \mathbb{R}^3$ such that $p(0) = s_i$ and $p(1) = s_j$. The function

$$\det \circ J_\phi \circ p : [0, 1] \to \mathbb{R} \tag{3}$$

is continuous as it is a composition of continuous functions. Indeed, $\boldsymbol{\phi}$ is of class C^1 by hypothesis—i.e., its Jacobian is a continuous function. The determinant of a matrix is also a continuous function, and p is a continuous by construction.

Thus, (3) is a continuous function starting negative in 0 and ending positive in 1. It follows that it must necessarily be zero at some point in $\bar{s} \in (0, 1)$, thus confirming that violated hypothesis is necessary and concluding the proof. ■

E. Learning Transportation Maps: ϕ

The inference of the function is made in two steps: first, the affine transformation $\boldsymbol{\gamma}(\boldsymbol{x})$ is obtained, and then the nonlinear transformation $\boldsymbol{\psi}(\boldsymbol{\gamma}(\boldsymbol{x}))$ is fitted only on the residual error.

a) Affine 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.,

$$\boldsymbol{\gamma} : \mathbb{R}^3 \to \mathbb{R}^3 \text{ s.t. } \boldsymbol{\mathcal{T}} - \bar{\boldsymbol{\mathcal{T}}} = \boldsymbol{\gamma}(\boldsymbol{\mathcal{S}} - \bar{\boldsymbol{\mathcal{S}}}) \tag{4}$$

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

$$\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = (\boldsymbol{\mathcal{S}} - \bar{\boldsymbol{\mathcal{S}}})^\top (\boldsymbol{\mathcal{T}} - \bar{\boldsymbol{\mathcal{T}}}) \tag{5}$$

and the rotation matrix is defined as

$$\boldsymbol{A} = \boldsymbol{V}\boldsymbol{U}^\top. \tag{6}$$

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 that the transformation is a proper rotation matrix without any reflection; see [46] for more details. Hence, the linear transformation on any point in the space can be computed as

$$\boldsymbol{\gamma}(\boldsymbol{x}) = \boldsymbol{A}(\boldsymbol{x} - \bar{\boldsymbol{\mathcal{S}}}) + \bar{\boldsymbol{\mathcal{T}}}. \tag{7}$$

However, if $\det(\boldsymbol{\Sigma}) = 0$, i.e., $\boldsymbol{\Sigma}$ is not full rank, it implies that the rotation is not uniquely defined and is set to the identity by default. Fig. 2(c) shows a linear transformation of the source and a grid of points from the original space depicted in Fig. 2(b).

Having a global affine transform reduces the necessary complexity of the nonlinear function. In fact, the norm of the distance between the target and the roto-translated source is lower or equal to the norm of the distance between the target and the original source, i.e.,

$$||\boldsymbol{\mathcal{T}} - \boldsymbol{\gamma}(\boldsymbol{\mathcal{S}})|| \leq ||\boldsymbol{\mathcal{T}} - \boldsymbol{\mathcal{S}}||.$$

The proof is a direct consequence that the affine transformation transforms the original source to be as close as possible to the target, given the linear constraint.

b) Nonlinear transformation: After fitting the linear transformation of (7), the residual transformation is obtained by substituting the source, target points, and the fitted linear function in (2), obtaining that

$$\boldsymbol{\psi} : \mathbb{R}^3 \to \mathbb{R}^3 \text{ s.t. } \boldsymbol{\mathcal{T}} - \boldsymbol{\gamma}(\boldsymbol{\mathcal{S}}) = \boldsymbol{\psi}(\boldsymbol{\gamma}(\boldsymbol{\mathcal{S}})). \tag{8}$$

The nonlinear function $\boldsymbol{\psi}$ can be any nonlinear regressor, such as a Multilayer Perceptron, a Gaussian Process (GP), or a Locally Weighted Translation (LWT) [31]. However, the inductive bias given by the nature of the nonlinear function will affect the regression output when transporting points far away from the source distribution. For example, suppose that the function is approximated with a GP with a distance-based kernel k, such as a square exponential kernel (see Appendix A). If the prior distribution is set to be a zero-mean function when making predictions in regions of the space far away from the source distribution points, the final transportation converges to just being an affine transformation, see Fig. 2(d). Knowing the out-of-distribution (o.o.d.) properties of our policy transportation is desirable, considering that we will transport points of the policy that are not necessarily close to the point of the source/target distribution. If no keypoints are provided, the zero deformation prior is applied to the space.

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

F. Transportation of Labels: Position and Velocity

This article focuses on generalizing the policy labels so that a new policy can be fitted to satisfy the new circumstances. Given our set of position labels $\mathcal{X}$ from the original policy, the transported labels $\hat{\mathcal{X}}$ are then obtained as

$$\hat{\mathcal{X}} = \phi(\mathcal{X}).$$

Although the transportation map allows the transport of any point of the original demonstration in the new situation, e.g., to generalize the demo on cleaning a new surface, we still have not formulated a transportation function for the velocity labels $\dot{\mathcal{X}}$. This 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 through feedback or aggregating new data from interactive demonstrations [1], [4]. 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 and by differentiating w.r.t. time on both sides and using the chain rule, we obtain the velocity labels in the transported space as

$$\dot{\hat{\mathcal{X}}} = \frac{\partial \phi(\mathcal{X})}{\partial \mathcal{X}} \dot{\mathcal{X}} = \boldsymbol{J}(\mathcal{X}) \dot{\mathcal{X}} \tag{9}$$

where the Jacobian matrix, using the definition of (2), can be defined as

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

where $\frac{\partial \boldsymbol{\gamma}(\boldsymbol{x})}{\partial \boldsymbol{x}} = \boldsymbol{A}$ and $\frac{\partial \boldsymbol{\psi}(\boldsymbol{x})}{\partial \boldsymbol{\gamma}(\boldsymbol{x})}$ 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 $\boldsymbol{x}$.

Figs. 3 and 4 summarize how the transportation function ϕ is used to transport the label of the original policy. We assume to have a set of labels used to fit the original dynamical system, i.e. $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$; we show how to transport these labels using the transportation function $\phi(\boldsymbol{x})$ before fitting the new dynamical system $g(\boldsymbol{x})$. If the original policy labels were sampled from a stable dynamical system, a sanity check on transportation is to ensure that the determinant of the Jacobian is larger than zero for any of the transported labels [34]; this also means that the function is locally invertible, hence locally diffeomorphic. If the map is diffeomorphic and the original labels were sampled from a stable dynamic system, they will maintain the stability property. However, given the new set of labels, if the task requires the motion to be asymptotically stable, then the proper regression [10], [8], [9], [31] can be selected to enforce this requirement (even if the whole labels do not preserve this property). When using a function approximator that allows setting a zero-prior and evaluating out of distribution, the Jacobian converges to the average rotation matrix, i.e., $\boldsymbol{J} \approx \boldsymbol{A}$. Fig. 4 shows the vector fields obtained by fitting f and g using a vanilla GP but different regressors can be chosen, like LPV [10] or NN [8].

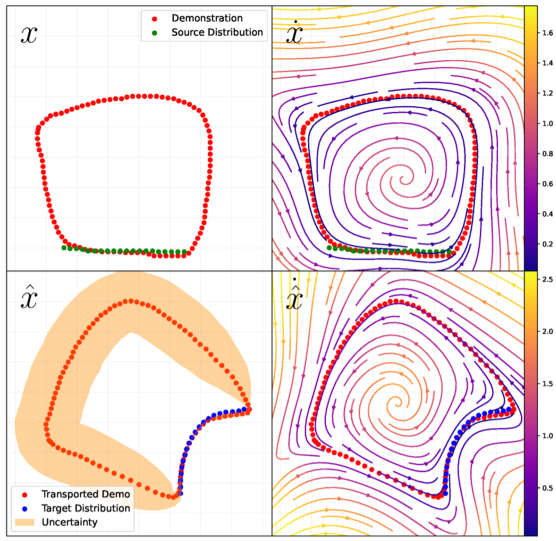


Fig. 4. Graphical representation of the mathematical scheme of policy transportation. The color temperature in the fields represents the uncertainty of the prediction.

G. Transportation of Labels: Orientation, Stiffness, and Damping

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, $\boldsymbol{R}$. The transportation of the tip of the vector can be obtained using the Taylor approximation of (2), according to

$$\hat{\boldsymbol{x}_{\text{tip}}} = \boldsymbol{\phi}(\boldsymbol{x}_{\text{tip}}) \approx \boldsymbol{\phi}(\boldsymbol{x_0}) + \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}} \boldsymbol{\epsilon} = \boldsymbol{\phi}(\boldsymbol{x_0}) + \boldsymbol{J}\boldsymbol{R}\boldsymbol{\epsilon}_0 \tag{10}$$

where $\boldsymbol{R} \in \mathbb{R}^{3\times 3}$ is the original robot orientation, $\boldsymbol{\epsilon}_0$ is a vector with infinitesimal dimension that has zero orientation.

Given a square matrix $\boldsymbol{J}$, the closest rotation matrix $\boldsymbol{J}_\perp \in \text{SO}(3)$ is found by projecting $\boldsymbol{J}$ onto the special orthogonal group $\text{SO}(3)$. The matrix $\boldsymbol{J}_\perp$ is obtained through the polar decomposition of $\boldsymbol{J}$ ensuring that $\det(\boldsymbol{J}_\perp) = +1$, thus $J_\perp$ is a proper rotation matrix within $\text{SO}(3)$.

From (10), it is readily apparent that the transported orientation labels of the robot end-effector become

$$\hat{\boldsymbol{\mathcal{R}}} = \boldsymbol{J}_\perp \boldsymbol{\mathcal{R}}. \tag{11}$$

In addition, when implementing policies on a Cartesian impedance control, the stiffness $\boldsymbol{K}$ 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.,

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

Hence, the generalization of the stiffness matrix becomes

$$\hat{\mathcal{K}} = \boldsymbol{J}_\perp \mathcal{K} \boldsymbol{J}_\perp{}^T$$

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

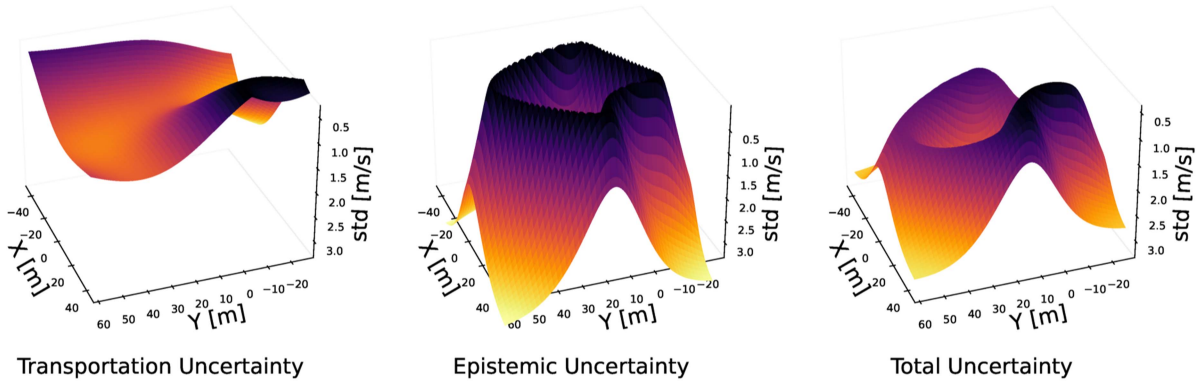


Fig. 5. Standard Deviation quantification on the velocity field. **Transportation Uncertainty** and quantifies the (heteroscedastic) uncertainty on the transported label (velocity) corresponding to the transported demonstration. **Epistemic Uncertainty** is the resulting model uncertainty when fitting the new policy $\hat{f}$. **Total Uncertainty** is the resulting standard deviation after computing the variance sum of transportation and epistemic uncertainties.

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

$$\hat{\mathcal{D}} = \boldsymbol{J}_{\perp} \mathcal{D} \boldsymbol{J}_{\perp}{}^{T}$$

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

H. Transporting Uncertainty

Given the reduced set of available keypoints, some parts of the space that are not close to them will be transported with a certain degree of uncertainty. A probabilistic model, like a GP or any approximator of its posterior like an ensemble neural network (E-NN) [47], will also provide the uncertainty on transported labels that are used to fit the new motion policy. In particular, a GP derivative is also a GP [48] and its existence will depend on the differentiability of the kernel function, see Appendix A. On the other hand, the uncertainty of the derivative in an E-NN is computed as the standard deviation of the derivatives of each model.

The uncertainty quantification of the transported labels becomes essential for calculating the final uncertainty on the control variable, e.g., the velocity. In Fig. 4, 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 velocity labels through the derivative of the (uncertain) transportation map of (9), i.e.,

$$\boldsymbol{\Sigma}_{\hat{\boldsymbol{x}}} = \boldsymbol{\Sigma}_{\frac{\partial \boldsymbol{\phi}(\boldsymbol{x})}{\partial \boldsymbol{x}}} \dot{\boldsymbol{x}}^{2} \tag{12}$$

given the definition of the weighted sum of Gaussian variables [49]. Hence, considering that the labels are uncertain, the prediction of the resulting policy can be computed as the sum of the epistemic (due to the reduced set of policy labels) and aleatoric uncertainty (due to the reduced set of keypoints), i.e.,

$$\boldsymbol{\Sigma}_{\dot{\hat{x}}} = \boldsymbol{\Sigma}_{\hat{f}} + \boldsymbol{\Sigma}_{\hat{\boldsymbol{x}}}. \tag{13}$$

Fig. 5 depicts the *transportation uncertainty* on the norm of the velocity and the epistemic uncertainty of the model $\hat{\boldsymbol{f}}$ using transported position and transported velocities labels. From Figs. 4 and 5, 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. 5 grows either when we go far away from the (transported) demonstration or away from the points of the source/target distribution.

IV. 2-D Simulations and Comparisons

The availability of (calibrated) uncertainties is an important feature that improves trustworthiness in deploying robot motion generalization. In this section, we evaluate different maps, for example, defining a Gaussian Process Transportation (GPT) for generalizing the demonstration in a 2-D surface cleaning task and on a multiple reference motion generalization. Our goal for these simulated experiments is the following:

1) to illustrate and compare how the generalization process differs when employing regressors other than a GP or methodologies from the state-of-the-art while generalizing a cyclic demonstration that approaches and then retreats from the surface "to clean," in Section IV-A;
2) assess and compare the policy transportation ability to generalize in multireference frame tasks, measuring its performance against state-of-the-art algorithms, in Section IV-B.

A. 2-D Surface Cleaning

Fig. 4 visualizes the transportation of the given demonstration, in red, from the source to the target space, using the transportation map ϕ where the nonlinear 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 nonlinear transportation. Table I summarises the method with their properties, while

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

TABLE I
SUMMARY TABLE OF DIFFERENT METHODS USED TO TRANSPORT TRAJECTORIES TO DIFFERENT SURFACES

Method	Modality	Vel. Transport	Diffeomorphic	Uncertainty
Reshaped-KMP	linear transform + trajectory-reshaping	✗	✗	analytical
LE [16]	linear transform + trajectory-reshaping	✗	✗	✗
LTW [31] [33]	linear transform + bijective map	✓	✓	✗
E-Flow [32], [50]	linear transform + bijective map	✓	✓	estimated
E-NN [51]	linear transform + residual map	✓	✗	estimated
GPR	transform + residual map	✓	✗	analytical

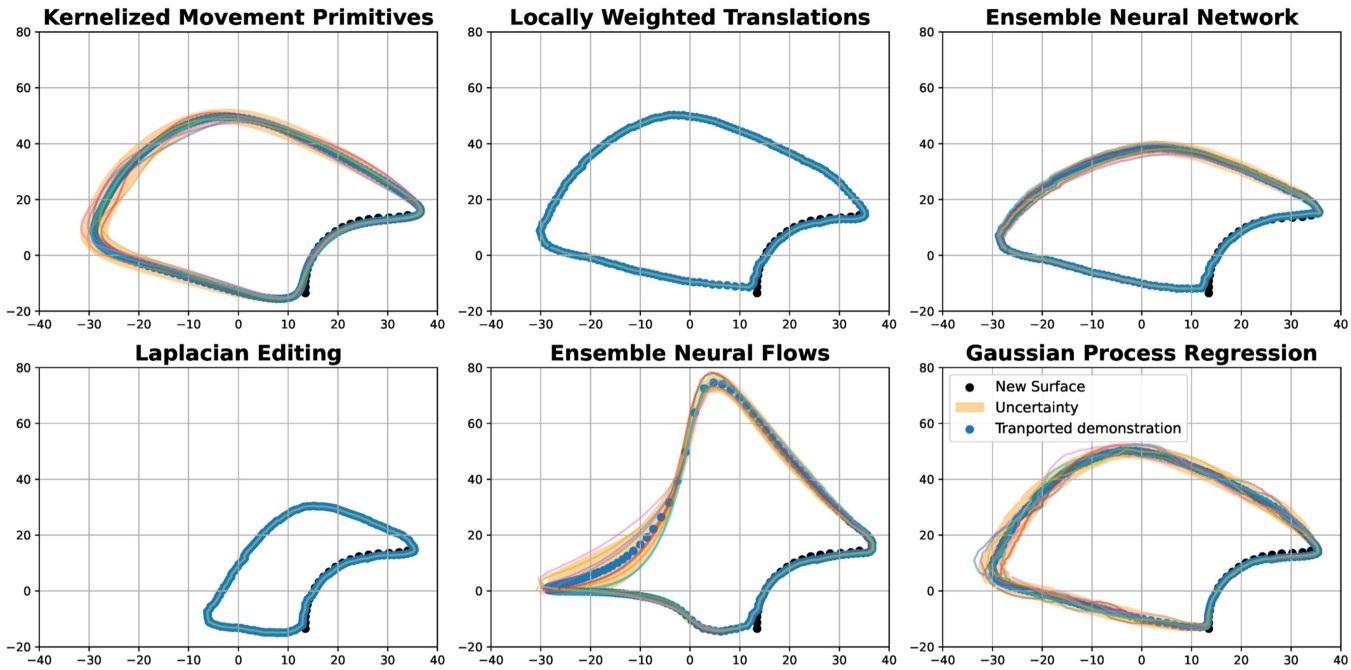


Fig. 6. Qualitative comparison of the transportation of a demonstration in target space for 2-D surface cleaning. The colored lines are the samples of the final transported trajectory policy, i.e. $\hat{x} = \phi(x)$, and the orange area is two standard deviations. The black curve is the 1-D surface to clean.

Fig. 6 shows the generalization of the demonstration when these methods are used. The illustrated trajectories are the result of the transportation of the original demonstration.

The figure is organized in the following way:

1) the first column shows the result of trajectory-based reshaping methods, KMP and LE;
2) the second column shows the result by using diffeomorphic maps;
3) the third column shows the results by using generic regression models, i.e., NN, and GPs.

KMP [7], in this study, fits the motion as a function of time, i.e. $\mathcal{X} = f(t)$, while LE [16], [17] 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. 6. For both of these trajectory-based methods, every point of the source distribution is matched with the closest point of the demonstration, solving the Hungarian assignment problem. Then, each point of the trajectory, or the graph, is moved, knowing the new desired target location of the matched points by using LE or the time-based equivalent for the transportation process, which we define as Reshaped-KMP, and it is governed by the following equations:

$$\hat{\mathcal{X}} = \mathcal{X} + \phi(\boldsymbol{t}), \quad \text{where} \quad \phi(\boldsymbol{t}_*) = \mathcal{T} - \mathcal{S}$$

where $\boldsymbol{t}_*$ corresponds to the time indexes in the trajectory of the selected elements to modify, and $\boldsymbol{t}$ is the complete set of time indexes of the recorded trajectory.

Neural Flows [52] are bijective neural networks, i.e., flows, usually used to learn a mapping from a simple probability distribution to a more complex target distribution, and it enforces $\psi(\boldsymbol{x})$ to be a diffeomorphism; similarly a Locally Weighted Translation (LWT) [31] also learn a diffeomorphism by consequently applying a local deformation of the space that would preserve the invertibility of the resulting total map. In order to estimate the uncertainties of the generalization, we train an ensemble of multiple individual models, trained independently, whose combined predictions are used to estimate a distribution on the prediction, i.e., mean and variance. Fig. 6 depicts the mean and the uncertainty bounds of 2-σ for the transported trajectories when using ensembles and GPs. The bounds are computed analytically for the GP from (14) and (15) and the

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

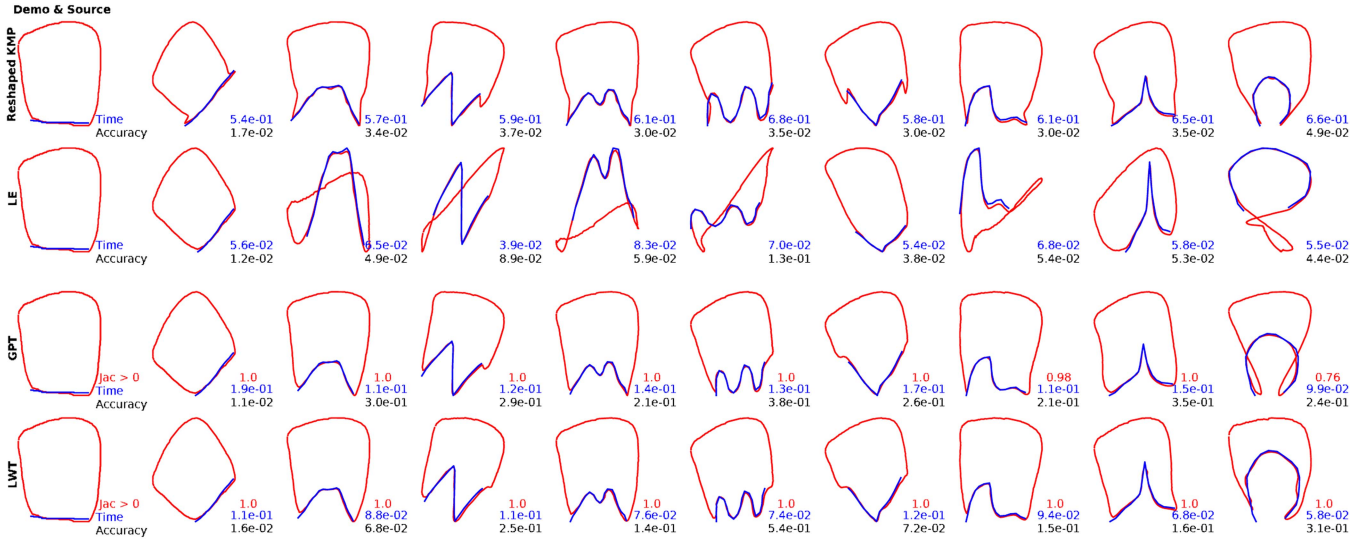


Fig. 7. Comparisons using many more surfaces. The first two rows used trajectory reshaping methods, while the last two used the proposed transportation method with a GP and an LWT. We also report the computational time [s] to fit and transform the regressor, the accuracy in matching overlapping source points with target points, and when using transportation maps, the percentage of transported points with a Jacobian larger than zero.

depicted GP samples of Fig. 6 are drawn from the posterior distribution.

From Fig. 6, the reader can appreciate how the GP-based methods are the only regressor with well-calibrated and unbiased epistemic uncertainty quantification and minimal mean 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 trajectory, even though the points are at the same distance from the surface. The Neural Flows show an undesired distortion of the demonstration when evaluating far from the surface. Differently, the LWT performs an undistorted generalization but does not provide any measure of uncertainty.

Quantitative Analysis: Fig. 7 illustrates how well different methods generalize a demonstrated trajectory across various surfaces. The methods compared include two trajectory reshaping techniques and two transportation methods, GPT and LWT, selected for their low out-of-distribution (o.o.d.) distortion. The figure shows that while Laplacian Editing often leads to highly distorted generalized trajectories, GPT and LWT perform similarly in terms of accuracy and computation time. For highly distorted surfaces (last column), some transported labels for the GP have a Jacobian determinant below zero, as evidenced by overlapping sections in the transported trajectory.

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 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., $\boldsymbol{\Delta x}_i = \boldsymbol{f}(\boldsymbol{x}_i)$ [53] [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 $\boldsymbol{\alpha_i}$, selects the mean and variance that can be used in a tracking algorithm, such as a Linear Quadratic Regulator [2], [54]. 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 [54], 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.

Unlike these task-parameterized approaches, our proposed method does not track only the reference frame but also a set of points that are relevant to the starting and goal objects. To describe a frame in 2D, we need at least two nonoverlapping points per frame; however, the transportation method will also scale if we extract more points; hence in Fig. 8, we extracted 5 points are tracked w.r.t. each reference frame when using GPT. However, for the LE, we used only 2 points per frame, as the redundancy of extracted points led to ambiguity in the assignment process and decreased algorithm performance. One of the main perks of our proposed method is the ability to generalize any motion skill generated by even only one demonstration, unlike GMM-based methods, where, to capture a meaningful mixture

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

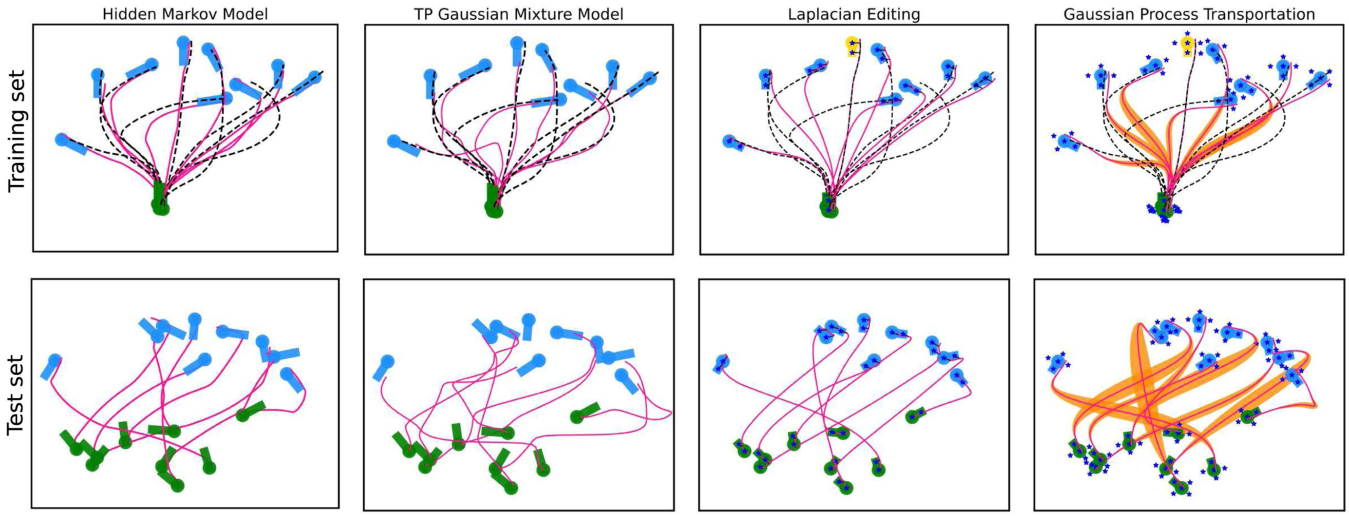


Fig. 8. Qualitative comparison of multireference frame parameterization. Comparison between HMM, TP-GMM, LEs, and our proposed method. The first row compares the performance in reproducing the training set demonstration, depicted as a dashed black line, where both HMM and TP-GMM are trained using all nine demonstrations, while LE 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 in LE and GPT 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. The yellow frame belongs to the source distribution.

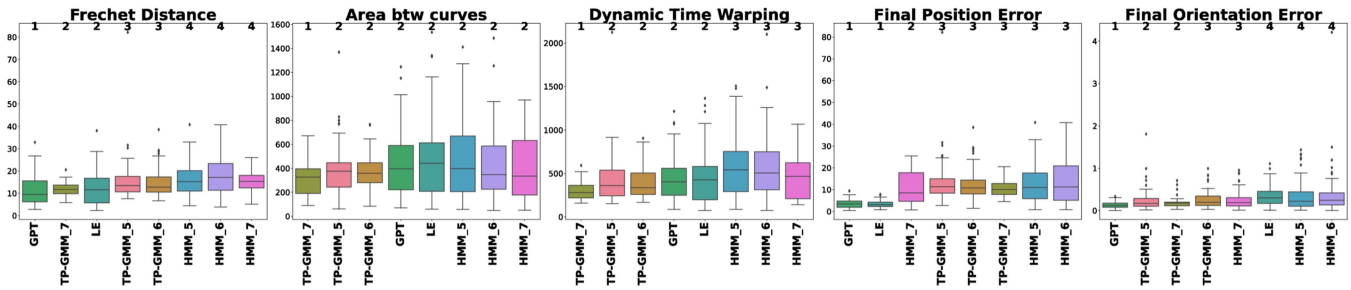


Fig. 9. Box plot results and performance ranking on frame configuration from the training set. The number on top of each box plot is the position of the method in the performance **ranking**. A higher value in the ranking (where the maximum is one) implies that the method has a more statistically significant discrepancy with another method than the next method in the ranking.

model, at least two diverse demonstrations need to be provided. In addition, the GMM uncertainty of the final multiframe 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. The illustrated trajectories (and uncertainties) both for GPT and LE are the result of the transportation of the single original demonstration.

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] [53] or as an optimal tracking problem of a multitransition HMM[2] [54]. However, when evaluating the test set generated on the random reorganization of the frames, the resulting trajectories do not successfully reach the goal frame either 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, LE, and GPT, on the training set. Nine demonstrations are available for different configurations of the starting and goal frames. 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 index, e.g., HMM_6 means that we used an HMM model with six demonstrations. When training LE or GPT, only one

[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/

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

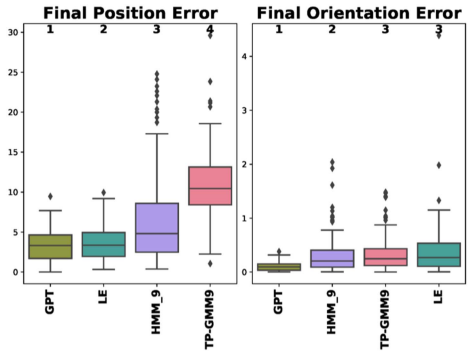


Fig. 10. Box plot results and performance ranking on randomly generated frame configurations.

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. They are as follows:

1) 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 [55];
2) area between the two curves that constructs quadrilaterals between two curves and calculates the area for each quadrilateral [55];
3) Dynamic Time Warping (DTW) that computes the cumulative distance between all points in the trajectory [55];
4) final position error, computed as the Euclidean distance between the final point of the trajectory and the rollout;
5) 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 Figs. 9 and 10. For this reason, we run a U test, also known as the Mann-Whitney nonparametric test [56], 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 number 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 between 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.

V. Real Robot Validation

To validate the proposed method on real manipulation tasks, we selected three challenging tasks, i.e., robot reshelving (Section V-A), dressing (Section V-B), and cleaning (Section V-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; e.g., 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 motion policy is learned with a nonparametric function approximation for motor learning that uses as input the position and (a belief of) time proposed in [12]. The desired attractor position, orientation, stiffness, damping, and time belief update are learned as a function of the current position-time input. 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/bE6uOnAQBLo.

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 as follows:

1) one global frame movement primitive is learned from a single demonstration and transported in the different object/goal configurations;
2) corner points of the objects and the shelf slot are tracked rather than position/orientation. To describe the 3-D pose of an object, we need at least two nonparallel vectors; hence, we need at least three nonoverlapping and noncollinear points.

Fig. 11 depicts the experimental setup where a milk box, with an AprilTag [57] on it, has to be positioned on a compartment on a shelf, also marked by another tag. The tags are a handy way in robotics setups to track keypoints of objects. However, the source of the keypoint can be obtained with different vision models like [28] and [58] without loss of generality for our proposed method. Before the demonstrations or execution, the robot searches for any tag in the spaces using the camera attached to its end-effector. For every tag, the transportation policy extracts a cube's center and corners with predefined side dimensions as the markers. Fig. 11 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

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

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

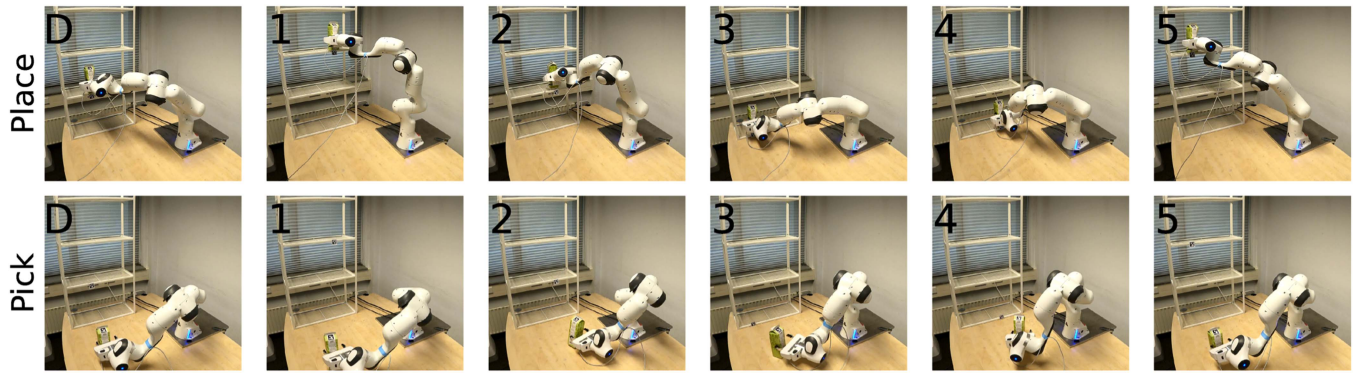


Fig. 11. Generalization of the reshelving task. The first column is the robot reproduction in the demonstration scenario.

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	–

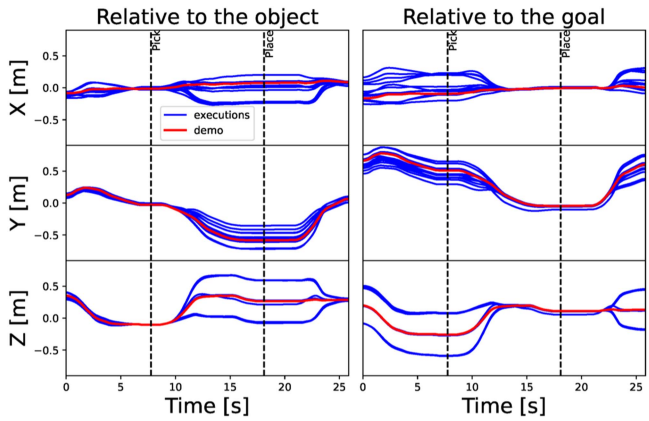


Fig. 12. Relative position of the end-effector w.r.t. the initial object and the goal position during multiple generalization rollouts in robot reshelving.

the ten different executions, while Fig. 12 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. 13 depicts the robot experimental setup where an articulated mannequin is posed in different shoulder positions and arm configurations. Four AprilTags [57] 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 center point is extracted. 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

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

The point cloud is obtained from a RealSense D435 camera. The user selects the relevant area, after which the points are smoothed along the z-axis and organized into a grid on x and y coordinates of dimension 20×20 where the corners are specified by a user input. This ensures that the source and target distributions have the same number of points, allowing for accurate matching and minimal estimation noise. 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 point cloud. The transportation between the

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

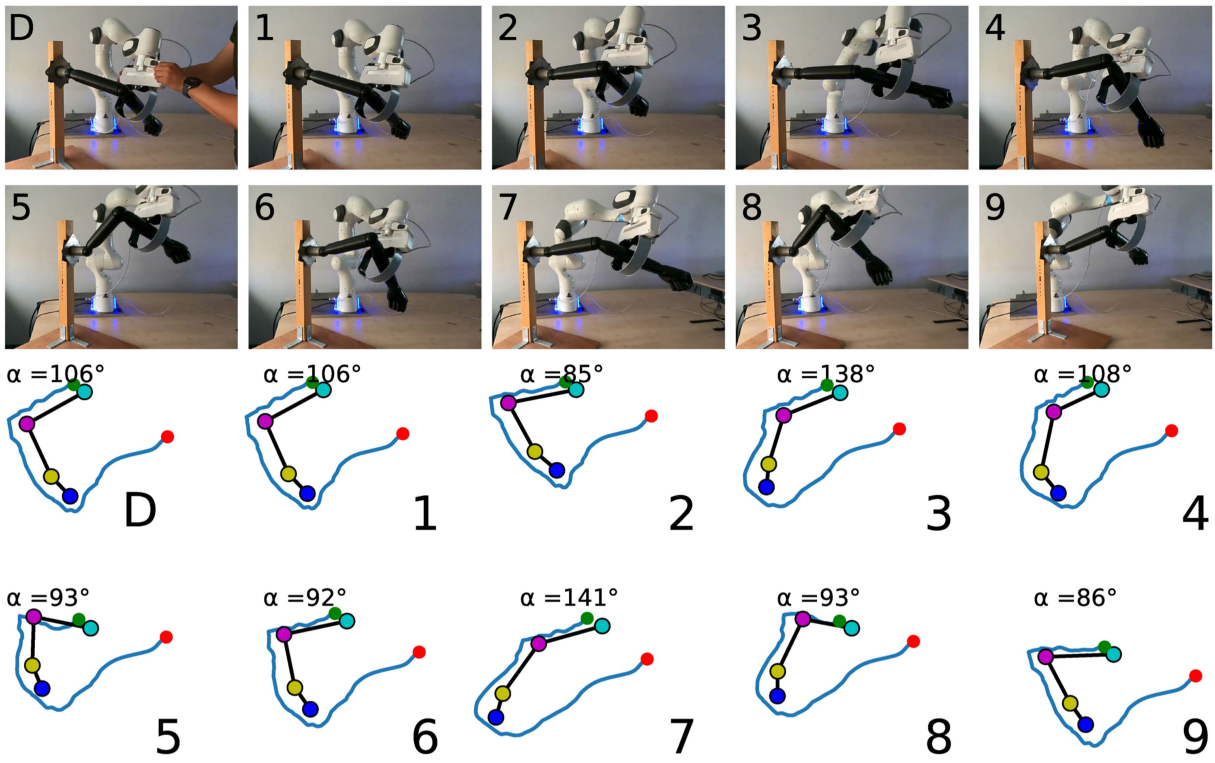


Fig. 13. Dressing policy generalization. Cyan marker is the shoulder, magenta is the elbow, yellow is the wrist, and blue is the hand. The blue rollout end-effector trajectory starts from the red dot and finishes with the green dot. α is the angle of the elbow; the smaller the angle, the more complicated the generalization would be.

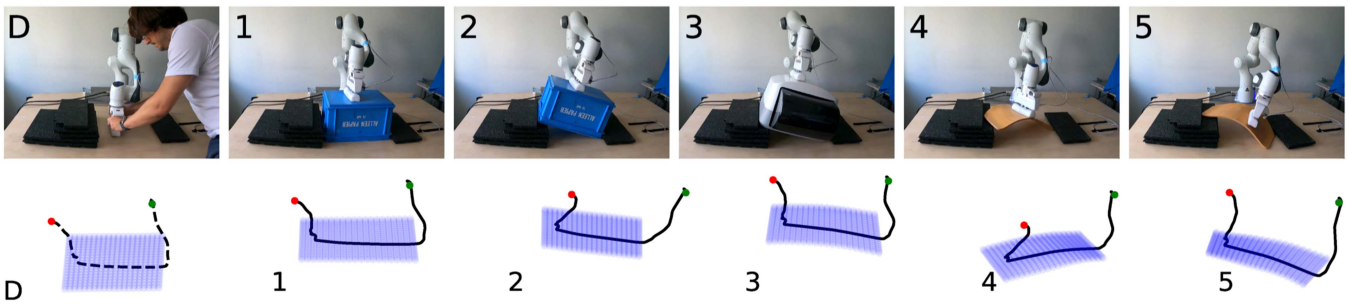


Fig. 14. Point cloud, demonstration and rollouts of the generalized motion in cleaning tasks.

(smoothened) source and the (smoothened) 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, see Appendix A.

Fig. 14 depicts the teaching of a cleaning task on a flat surface and the generalization on different heights, tilted, 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 rollout transported policy (depicted a solid line). Fig. 14 also highlights how the rollouts 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 to 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. 15 depicts the estimated norm of the force exchanged with the surface, where the same increasing/decreasing trend is captured on the different surfaces.

D. Towards Policy Transportation Using DINO Features

Relying on keypoints extracted from fiducial markers is not always practical in real-world applications. Nevertheless,

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

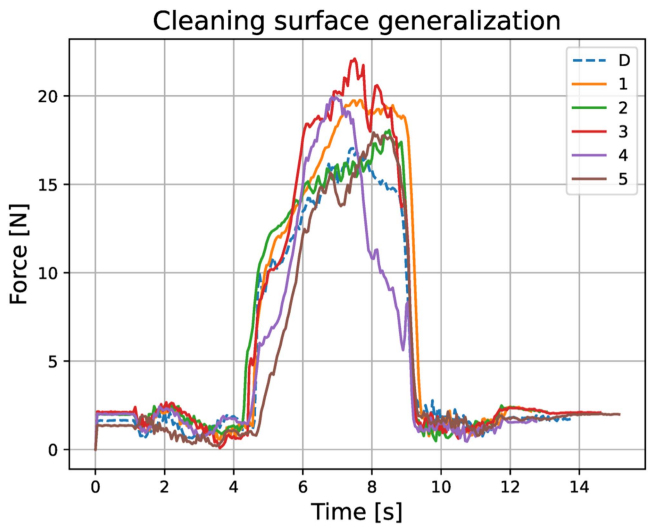


Fig. 15. Norm of the force perceived [N] from the end-effector when executing the transported dynamics on the new surfaces.

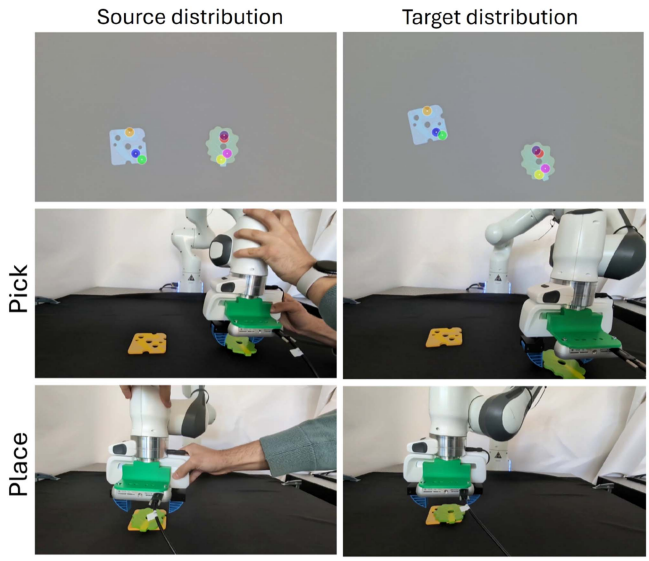


Fig. 16. Food Stacking Task. The single demo is displayed on the left, and the motion generalization is on the right. The source and the target keypoints are automatically detected using DINO features correspondences [59]. The 3-D location of the keypoint correspondence is obtained using the stereoscopic depth map.

given the raw camera input and an estimated depth, we can leverage new foundation models to automatically extract and find the corresponding features [59]. This experiment is similar to what was shown in DINObot [28]. However, we are not only roto-translating the policy, but we also apply the nonlinear transformation proposed in this article. Fig. 16 shows the source, recorded before giving the demonstration, and the target image on a new out-of-distribution configuration of the objects. A stereoscopic depth map is used to estimate the 3-D location of each feature, so as to generate the necessary keypoints to perform the generalization. We ask the model to find a maximum of 7 keypoints, and a threshold of 0.2 was applied to the saliency maps, retaining only features with higher attention values; this avoids finding features on the table. We moved the objects around the workspace, always with the (toy) cheese on the left and the (toy) lettuce on the right, and used a transportation policy with a GP to perform the generalization of the original position and orientation labels. Despite very successful results in generalizing in 10 different configurations, we identify poor orientation generalization when swapping the location of the objects, i.e., lettuce on the left and cheese on the right side of the table. On the computational side, the main bottleneck is the DINO feature extraction, which, on the current implementation [59] on an RTX4060, can take almost a minute to find the keypoint correspondence. Moreover, the depth estimation turned out to be the brittle component of the pipeline when working with objects that are not flat.

VI. 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 preprocessing algorithms, such as optimal transport [60] or iterative closest point, or Coherent Point Drift [42] need to be adopted to perform the matching. In addition, semantic matching can increase cross-domain generalization, e.g., by adapting the reshelving strategy to a completely different shelf type or adjusting the dressing policy from an adult to a baby arm or to manipulate different food shapes or reshelving boxes of different dimensions.

Moreover, in the current implementation and experimentation, we assume full observability of the keypoints and can estimate their location with high accuracy. We foresee exciting directions in dealing with partial observability and different sensor noise, in particular when working with objects with very irregular shapes and estimating the keypoint location using the depth cameras.

Another assumption of the developed method is that it deals with static environments, i.e., the target distribution does not change during the policy's rollout. However, this assumption can fail when dealing with the reshelving of moving objects or when trying to dress real humans that will probably move before and during the interaction [61], [62]. 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 (2) 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 original policy labels and how. In addition, in case many source distributions/policies are recorded, selecting the best one, according to

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

some similarity to the current observed target distribution, can open exciting developments of scaling the proposed theory to work on a big database of keypoint parameterized demonstrations.

VII. Conclusion

In this article, 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. In addition, 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 transportation algorithm was tested with different state-of-the-art regressors and compared with different generalization methods, showing how, even with only one demonstration, it results in better or comparable motion generalization.

We validated the proposed approach on a Franka Robot, testing it on four different tasks: product reshelving, arm dressing, and surface cleaning, and table-top pick-and-place. These various tasks were never tackled together by the same generalization algorithm, and they were usually performed with ad hoc solutions, e.g., to keep a constant force when cleaning a surface. 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 a stereo camera. Future development will have to focus on scaling the manipulation generalization using large, unmatched, and partially observable point clouds of complex, deformable, and moving objects while allowing the use of human feedback in the fine-tuning of the resulting policy, asking for human attention when facing high-uncertainty generalization regions, or actively searching for missing keypoints that would reduce the total transportation uncertainty.

Appendix A
Gaussian Process Regression

A GP 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

$$\boldsymbol{\mu} = \boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}}(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{y} \tag{14}$$

$$\boldsymbol{\Sigma} = \boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}_*} - \boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}}(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}_*} \tag{15}$$

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, e.g., the squared exponential kernel defined as

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

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 [48]. In the context of this article, when dealing with multioutput GPs, we assume the same smoothness, prior noise, and likelihood noise among the models, hence we compute $(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma_n^2\boldsymbol{I})^{-1}$ once for all the three models and during inference the vector $\boldsymbol{K}_{\boldsymbol{X}_*,\boldsymbol{X}}$ is also the same for the three outputs.

Moreover, 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 pseudodata, 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 as the evidence of the data [63], [64].

The use of SVGP, in the context of this article, 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 this article is independent of the approximation nature of the GP.

When dealing with derivatives of transportation maps that are estimated with probabilistic functions, it is necessary to estimate the uncertainty over the predicted derivatives. Fortunately, a GP derivative is also a GP [48] 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 G become

$$\boldsymbol{\mu'} = \boldsymbol{K}^{10}_{\boldsymbol{X}_*,\boldsymbol{X}}(\boldsymbol{K}_{\boldsymbol{X},\boldsymbol{X}} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{y}$$

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

References

[1] C. Celemin et al., "Interactive imitation learning in robotics: A survey," *Found. Trends Robot.*, vol. 10, no. 1-2, pp. 1–197, 2022.

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

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

[3] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, "kPAM: Keypoint affordances for category-level robotic manipulation," in *Proc. Int. Symp. Robot. Res.*. Springer, 2019, pp. 132–157.

[4] G. Franzese, A. Mészáros, L. Peternel, and J. Kober, "ILoSA: Interactive learning of stiffness and attractors," in *Proc. 2021 IEEE/RSJ Int. Conf. Intell. Robots Syst.*. IEEE, 2021, pp. 7778–7785.

[5] T. Li and N. Figueroa, "Task transfer with stability guarantees via elastic dynamical system motion policies," in *Proc. 7th Annu. Conf. Robot Learn.*, 2023, pp. 1–33.

[6] M. Saveriano, F. J. Abu-Dakka, A. Kramberger, and L. Peternel, "Dynamic movement primitives in robotics: A tutorial survey," *Int. J. Robot. Res.*, vol. 42, no. 13, pp. 1133–1184, 2023.

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

[8] R. Pérez-Dattari and J. Kober, "Stable motion primitives via imitation and contrastive learning," *IEEE Trans. Robot.*, vol. 39, no. 5, pp. 3909–3928, Oct. 2023.

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

[10] N. B. Figueroa Fernandez and A. Billard, "A physically-consistent Bayesian non-parametric mixture model for dynamical system learning," in *Proc. Conf. Robot Learn.*. PMLR, 2018, pp. 927–946.

[11] Z. Brohan et al., "Rt-2: Vision-language-action models transfer web knowledge to robotic control," in *Proc. Conf. Robot Learn.*, 2023, pp. 2183–2183.

[12] G. Franzese, L. de Souza Rosa, T. Verburg, L. Peternel, and J. Kober, "Interactive imitation learning of bimanual movement primitives," *IEEE/ASME Trans. Mechatron.*, vol. 29, no. 5, pp. 4006–4018, Oct. 2024.

[13] M. Saveriano, F. Franzel, and D. Lee, "Merging position and orientation motion primitives," in *Proc. 2019 Int. Conf. Robot. Automat.*. IEEE, 2019, pp. 7041–7047.

[14] I. Nematollahi, K. Yankov, W. Burgard, and T. Welschehold, "Robot skill generalization via keypoint integrated soft actor-critic gaussian mixture models," in *Proc. Int. Symp. Exp. Robot.*, 2023, pp. 168–180.

[15] A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann, "Probabilistic movement primitives," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, vol. 26, pp. 2616–2624.

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

[17] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel, "Laplacian surface editing," in *Proc. 2004 Eurographics/ACM SIGGRAPH Symp. Geometry Process.*, 2004, pp. 175–184.

[18] M. Y. Seker, M. Imre, J. H. Piater, and E. Ugur, "Conditional neural movement primitives," in *Proc. Robot.: Sci. Syst.*, vol. 10, 2019, pp. 1–9.

[19] S. Calinon, F. Guenter, and A. Billard, "On learning the statistical representation of a task and generalizing it to various contexts," in *Proc. 2006 IEEE Int. Conf. Robot. Automat.*. IEEE, 2006, pp. 2978–2983.

[20] S. Calinon, F. Guenter, and A. Billard, "On learning, representing, and generalizing a task in a humanoid robot," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 37, no. 2, pp. 286–298, 2007.

[21] 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 *Proc. 9th IEEE-RAS Int. Conf. Humanoid Robots*. IEEE, 2009, pp. 582–588.

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

[23] Y. Huang, J. Silvério, L. Rozo, and D. G. Caldwell, "Generalized task-parameterized skill learning," in *Proc. 2018 IEEE Int. Conf. Robot. Automat.*. IEEE, 2018, pp. 5667–5474.

[24] A. Sena, B. Michael, and M. Howard, "Improving task-parameterised movement learning generalisation with frame-weighted trajectory generation," in *Proc. 2019 IEEE/RSJ Int. Conf. Intell. Robots Syst.*. IEEE, 2019, pp. 4281–4287.

[25] F. Khadivar, I. Lauzana, and A. Billard, "Learning dynamical systems with bifurcations," *Robot. Auton. Syst.*, vol. 136, 2021, Art. no. 103700.

[26] H. Perez-Villeda, J. Piater, and M. Saveriano, "Learning and extrapolation of robotic skills using task-parameterized equation learner networks," *Robot. Auton. Syst.*, vol. 160, 2023, Art. no. 104309.

[27] J. Yang, Z.-A. Cao, C. Deng, R. Antonova, S. Song, and J. Bohg, "Equibot: Sim (3)-equivariant diffusion policy for generalizable and data efficient learning," in *Proc. 8th Annu. Conf. Robot Learn.*, 2024. [Online]. Available: https://openreview.net/forum?id=ueBmGhLOXP

[28] N. Di Palo and E. Johns, "DINOBot: Robot manipulation via retrieval and alignment with vision foundation models," in *Proc. 2024 IEEE Int. Conf. Robot. Automat.*, 2024, pp. 2798–2805.

[29] W. Huang, C. Wang, Y. Li, R. Zhang, and L. Fei-Fei, "ReKep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation," in *Proc. 8th Annu. Conf. Robot Learn.*, 2024. [Online]. Available: https://openreview.net/forum?id=9iG3SEbMnL

[30] S. Patel et al., "A real-to-sim-to-real approach to robotic manipulation with VLM-generated iterative keypoint rewards," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2025. [Online]. Available: https://ras.papercept.net/conferences/conferences/ICRA25/program/ICRA25_ContentListWeb_2.html

[31] N. Perrin and P. Schlehuber-Caissier, "Fast diffeomorphic matching to learn globally asymptotically stable nonlinear dynamical systems," *Syst. Control Lett.*, vol. 96, pp. 51–59, 2016.

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

[33] X. Gao, J. Silvério, E. Pignat, S. Calinon, M. Li, and X. Xiao, "Motion mappings for continuous bilateral teleoperation," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 5048–5055, Jul. 2021.

[34] G. Balakrishnan, A. Zhao, M. R. Sabuncu, J. Guttag, and A. V. Dalca, "VoxelMorph: A learning framework for deformable medical image registration," *IEEE Trans. Med. Imag.*, vol. 38, no. 8, pp. 1788–1800, Aug. 2019.

[35] A. Mészáros, G. Franzese, and J. Kober, "Learning to pick at non-zero-velocity from interactive demonstrations," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 6052–6059, Jul. 2022.

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

[37] E. Pignat and S. Calinon, "Learning adaptive dressing assistance from human demonstration," *Robot. Auton. Syst.*, vol. 93, pp. 61–75, 2017.

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

[39] W. Amanhoud, M. Khoramshahi, M. Bonnesoeur, and A. Billard, "Force adaptation in contact tasks with dynamical systems," in *Proc. 2020 IEEE Int. Conf. Robot. Automat.*. IEEE, 2020, pp. 6841–6847.

[40] L. Armesto, J. Moura, V. Ivan, M. S. Erden, A. Sala, and S. Vijayakumar, "Constraint-aware learning of policies by demonstration," *Int. J. Robot. Res.*, vol. 37, no. 13-14, pp. 1673–1689, 2018.

[41] N. Courty, R. Flamary, A. Habrard, and A. Rakotomamonjy, "Joint distribution optimal transportation for domain adaptation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, vol. 30, pp. 3733–3742.

[42] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 12, pp. 2262–2275, Dec. 2010.

[43] H. Tsukamoto, S.-J. Chung, and J.-J. E. Slotine, "Contraction theory for nonlinear stability analysis and learning-based control: A tutorial overview," *Annu. Rev. Control*, vol. 52, pp. 135–169, 2021.

[44] K. Kronander, M. Khansari, and A. Billard, "Incremental motion learning with locally modulated dynamical systems," *Robot. Auton. Syst.*, vol. 70, pp. 52–62, 2015.

[45] C. K. Fourie, N. Figueroa, and J. A. Shah, "On-manifold strategies for reactive dynamical system modulation with non-convex obstacles," *IEEE Trans. Robot.*, vol. 40, pp. 2390–2409, 2024.

[46] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-9, no. 5, pp. 698–700, Sep. 1987.

[47] A. G. Wilson and P. Izmailov, "Bayesian deep learning and a probabilistic perspective of generalization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 4697–4708.

[48] C. K. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*, vol. 2. Cambridge, MA, USA: MIT Press, 2006.

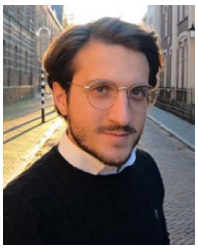
[49] M. P. Deisenroth, A. A. Faisal, and C. S. Ong, *Mathematics for Machine Learning*. Cambridge, U.K.: Cambridge Univ. Press, 2020.

[50] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real NVP," in *Proc. Int. Conf. Learn. Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=HkpbnH9lx

[51] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, vol. 30, pp. 6405–6416.

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.

[52] I. Kobyzev, S. J. Prince, and M. A. Brubaker, "Normalizing flows: An introduction and review of current methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 11, pp. 3964–3979, Nov. 2021.
[53] T. Alizadeh and B. Saduanov, "Robot programming by demonstration of multiple tasks within a common environment," in *Proc. 2017 IEEE Int. Conf. Multisensor Fusion Integration Intell. Syst.*. IEEE, 2017, pp. 608–613.
[54] S. Calinon, A. Pistillo, and D. G. Caldwell, "Encoding the time and space constraints of a task in explicit-duration hidden Markov model," in *Proc. 2011 IEEE/RSJ Int. Conf. Intell. Robots Syst.*. IEEE, 2011, pp. 3413–3418.
[55] 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," *Int. J. Mater. Forming*, vol. 12, pp. 355–378, 2019.
[56] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *Ann. Math. Statist.*, vol. 18, pp. 50–60, 1947.
[57] J. Wang and E. Olson, "AprilTag 2: Efficient and robust fiducial detection," in *Proc. 2016 IEEE/RSJ Int. Conf. Intell. Robots Syst.*. IEEE, 2016, pp. 4193–4198.
[58] I. Nematollahi, E. Rosete-Beas, A. Röfer, T. Welschehold, A. Valada, and W. Burgard, "Robot skill adaptation via soft actor-critic Gaussian mixture models," in *Proc. 2022 Int. Conf. Robot. Automat.*. IEEE, 2022, pp. 8651–8657.
[59] S. Amir, Y. Gandelsman, S. Bagon, and T. Dekel, "Deep ViT features as dense visual descriptors," 2021, *arXiv:2112.05814*.
[60] Y. Luo, Z. Jiang, S. Cohen, E. Grefenstette, and M. P. Deisenroth, "Optimal transport for offline imitation learning," in *Proc. 11th Int. Conf. Learn. Representations*, 2023, pp. 1–17.
[61] S. Li, N. Figueroa, A. Shah, and J. Shah, "Provably safe and efficient motion planning with uncertain human dynamics," *Robot.: Sci. Syst.*, 2021. [Online]. Available: https://doi.org/10.15607/RSS.2021.XVII.050
[62] J. Zhu, M. Gienger, G. Franzese, and J. Kober, "Do you need a hand?—A bimanual robotic dressing assistance scheme," *IEEE Trans. Robot.*, vol. 40, pp. 1906–1919, 2024.
[63] M. Titsias, "Variational learning of inducing variables in sparse gaussian processes," in *Proc. 12th Int. Conf. Artif. Intell. Statist.*. PMLR, 2009, vol. 5, pp. 567–574.
[64] J. Hensman, N. Fusi, and N. D. Lawrence, "Gaussian processes for Big Data," in *Proc. 29th Conf. Uncertainty Artif. Intell.*, 2013, pp. 282–290.

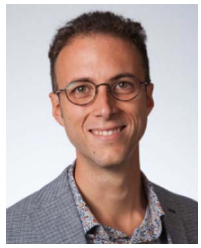


Giovanni Franzese received the B.Sc. degree in mechanical engineering and the M.Sc. degree in mechatronics and robotics from Politecnico di Milano, Italy, in 2016 and 2018, respectively, and the Ph.D. degree in robotics from the Department of Cognitive Robotics, TU Delft, Delft, The Netherlands, in 2025.

He is an ELLIS Member for artificial intelligence. His research interests include Interactive Imitation Learning applied to robot manipulation.

Ravi Prakash (Member, IEEE) received the Ph.D. degree in control and automation from the Department of Electrical Engineering, Indian Institute of Technology Kanpur, India, in 2022.

He is currently an Assistant Professor with the Department of Cyber Physical Systems (CPS), IISc Bangalore, India. Prior to this, he was a Postdoctoral Researcher with the Learning and Autonomous Control group, Department of Cognitive Robotics, TU Delft, Delft, The Netherlands. He is a DAAD AInet Postdoc Fellow for AI and Robotics.



Cosimo Della Santina (Senior Member, IEEE) received the Ph.D. degree (cum laude) in robotics from the University of Pisa, Italy, in 2019.

He is currently an Associate Professor with TU Delft, Delft, The Netherlands, and a Research Scientist with the German Aerospace Institute (DLR). From 2017 to 2019, he was a Visiting Ph.D. student and a Postdoc with the CSAIL, Massachusetts Institute of Technology, Cambridge, MA, USA. His research interest includes providing motor intelligence to physical systems, focusing on elastic and soft robots.

Dr. Della Santina was the recipient of the Georges Giralt Ph.D. Award in 2020, the IEEE RAS Early Academic Career Award in 2023, an ERC StG, and an NWO VENI.

Jens Kober (Senior Member, IEEE) received the Ph.D. degree in engineering from TU Darmstadt, Darmstadt, Germany, and the MPI for Intelligent Systems, Stuttgart, Germany, in 2012.

He is currently an Associate Professor with the TU Delft, Delft, The Netherlands. He worked as a Postdoctoral Scholar jointly with the CoR-Lab, Bielefeld University, Germany, and with the Honda Research Institute Europe, Germany.

Dr. Kober was the recipient of the annually awarded Georges Giralt Ph.D. Award for the best Ph.D. thesis in robotics in Europe for his research, the 2018 IEEE RAS Early Academic Career Award, the 2022 RSS Early Career Award, and an ERC Starting grant.

Authorized licensed use limited to: TU Delft Library. Downloaded on August 07,2025 at 07:23:47 UTC from IEEE Xplore. Restrictions apply.