

Learning from Multiple Demonstrations using Trajectory-Aware Non-Rigid Registration with Applications to Deformable Object Manipulation

Alex X. Lee

Abhishek Gupta

Henry Lu

Sergey Levine

Pieter Abbeel

Abstract—Learning from demonstration by means of non-rigid point cloud registration is an effective tool for learning to manipulate a wide range of deformable objects. However, most methods that use non-rigid registration to transfer demonstrated trajectories assume that the test and demonstration scene are structurally very similar, with any variation explained by a non-linear transformation. In real-world tasks with clutter and distractor objects, this assumption is unrealistic. In this work, we show that a trajectory-aware non-rigid registration method that uses multiple demonstrations to focus the registration process on points that are relevant to the task can effectively handle significantly greater visual variation than prior methods that are not trajectory-aware. We demonstrate that this approach achieves superior generalization on several challenging tasks, including towel folding and grasping objects in a box containing irrelevant distractors.

I. INTRODUCTION

Learning from demonstration has emerged as a powerful and effective framework for teaching robots to perform complex motion skills. A key challenge in learning from demonstration for robotic manipulation is to transfer a motion that is demonstrated on one object to another one. When the manipulation is performed on a deformable object, such as a rope or a towel, this transfer problem can be especially challenging, since the demonstration must be warped via a complex and nonlinear transformation to conform to the object’s shape. Furthermore, in order for a trajectory transfer method to generalize to a wide range of tasks without extensive hand-engineering, the transfer procedure must be automatic and general.

One practical and general approach to perform trajectory transfer for deformable object manipulation is to register a point cloud of the current object to the object used in the demonstration, determine the transformation function that maps one object to the other, and transform the demonstrated trajectory by the same function. This approach has been shown to produce effective behaviors for tasks such as knot tying [1], towel folding [2], [3] and simplified surgical suturing [4]. These approaches register point clouds of the current scene to the demonstration scene, typically using a color filter or another masking technique to pick out points of interest. However, real-world scenes exhibit considerable variation in shape and color, even when the salient components of the manipulated objects change only slightly, making it highly

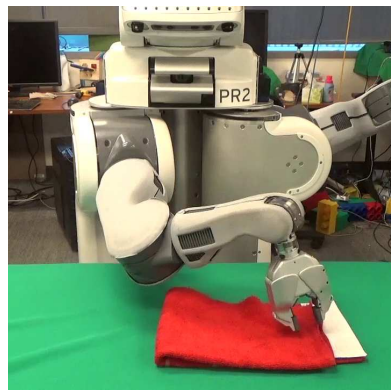


Fig. 1: A PR2 learned to fold a towel from human demonstrations.

nontrivial to identify or even define the points of interest. If we can inform the registration process about which points are actually relevant to the task, we can greatly improve the robustness and generalization power of registration-based trajectory transfer.

In this work, we show that such task-aware registration can be performed by including the demonstrated trajectories in the registration process, instead of computing the transformation entirely from the point clouds and only then applying it to the trajectories. Our trajectory-aware non-rigid registration method registers multiple demonstration point clouds to the current scene, computing the transformations that best put the demonstrated trajectories into agreement. This agreement is quantified by means of the variance of time-aligned points along the trajectories. In this approach, a good transformation is one that puts all of the demonstrated trajectories close together, even if this means that some points in the point clouds are not aligned well. This allows our method to effectively handle distractor objects and irrelevant variations in the object geometry.

Our main contribution is a trajectory-aware non-rigid registration method that registers multiple point clouds, together with their trajectories, to the current scene. We experimentally demonstrate that this approach outperforms standard trajectory transfer methods based on non-rigid registration that is not trajectory aware, especially in the presence of irrelevant distractor objects or when a significant part of the object being manipulated is irrelevant for the task. We present experimental results on several challenging tasks including towel folding and grasping objects in a box containing irrelevant distractors.

Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, CA, USA. {alexlee-gk, abhigupta, armentai, sergey.levine, pabbeel}@berkeley.edu

II. RELATED WORK

Learning from demonstration, also known as programming by demonstrations, has emerged as a practical and effective approach for specifying complex robotic manipulation skills [5], [6], [7]. One of the principal challenges in learning from demonstration for robotic manipulation is to adapt the demonstrated trajectory to the configuration of the objects at test-time. This adaptation is typically done by means of features on the manipulated objects [8], [9]. However, designing features can be challenging and time-consuming.

Several previous approaches have used warping to transfer manipulations between objects. One approach transfers grasps from a known object onto a new one by finding a mapping of contact points between the objects [10], [11]. Skill transfer between objects has also been done by using deformable registrations of RGB-D images [12]. These approaches focus on finding a registration between single objects and then using that registration to warp the pose or the finger position of the manipulator around the surface of the novel object.

Instead of warping only the points on the surface of the object, Schulman et al. [1] use non-rigid registration to compute a warping function for the entire scene. Our trajectory transfer approach builds on this work. In this approach, point clouds obtained from a depth sensor are used to adapt the trajectory. The point cloud in the demonstration is registered to the current test scene, and a transformation function is constructed from this registration that transforms demonstration points into the test scene. This transformation is then applied to the demonstrated trajectory. This work has also been extended to jointly optimize the transformation function and the trajectory in a unified optimization [13] and to incorporate normals to find better registrations [3]. These approaches are effective at aligning the trajectory with the current object. However, the underlying assumption of these methods is that the test scene is structurally similar to the demonstration scene. This assumption is reasonable when both scenes consist, for example, of a single deformable object. However, when the scenes have multiple independent parts, this method fails to differentiate between parts that are more or less salient for generalization.

In order to determine which parts of the scene are most relevant for the task, we use trajectories from multiple demonstrations, and choose the registration that puts all of these trajectories into alignment with the current test scene. The idea of using multiple demonstrations to improve transfer and generalization has been explored in the context of autonomous flight [14], autonomous driving [15], and dynamic movement primitives [16]. In the context of deformable object manipulation, multiple demonstrations have also been used to recover variable impedance control policies that trade off force and position errors [2]. However, to the best of our knowledge, ours is the first method that incorporates information from multiple demonstrated trajectories into the objective for non-rigid registration, with the goal of enabling computing a registration that is most appropriate to the task.

III. PRELIMINARIES

In this section, we review Schulman et al.’s approach [1] for trajectory transfer, the coherent point drift (CPD) algorithm [17] for point set registration, and the thin plate spline (TPS) [18], [19] parametrization for non-rigid transformations. Although the topics in this section are not new, their combination is novel to this work and forms the basic building blocks for the algorithm presented in this paper, which combines these approaches together with trajectory-aware registration in a unified probabilistic framework.

A. Learning from Demonstrations via Trajectory Transfer

In the method proposed by Schulman et al. [1], a demonstration consists of a point cloud \mathbf{X} of the demonstration scene and a sequence of end-effector poses. At test time, a test point cloud \mathbf{Y} is observed. The goal is to generalize the demonstrated trajectory to the new scene. Schulman et al. use the TPS-RPM algorithm [20] to find a non-rigid registration that maps points from the demonstration scene to the new test scene. Then, the registration function is applied to the demonstration trajectory to get a warped trajectory. This has the effect that the resulting end-effector motion incorporates variations in the new scene, which is important in manipulation tasks. The resulting trajectory does not incorporate collision avoidance and joint limits, so trajectory optimization [21] is then used to find a feasible joint angle trajectory.

When a collection of K demonstrations is available, $\mathcal{D} = \{\mathcal{D}^1, \dots, \mathcal{D}^K\}$, a registration function that maps points from the demonstration scene \mathbf{X}^k to the test scene \mathbf{Y} is independently computed for *each* demonstration \mathcal{D}^k , and the demonstration that incurs the least registration cost is chosen for trajectory transfer. In this work, we will instead use all of the demonstrations together, as discussed in Section IV. The prior method also makes use of the TPS-RPM algorithm for registration [1]. In this work, we instead use the CPD algorithm, described in the following subsection, which provides a substantial improvement in outlier handling, as shown in our experiments.

B. Coherent Point Drift

The CPD algorithm finds a registration between a source and target point set, which in our case are point clouds from a depth camera. Denote the source point $\mathbf{X} = [\mathbf{x}_1 \ \dots \ \mathbf{x}_N]^\top$ and the target point set $\mathbf{Y} = [\mathbf{y}_1 \ \dots \ \mathbf{y}_M]^\top$, with points $\mathbf{x}_i, \mathbf{y}_j \in \mathbb{R}^D$ for some dimension D . The registration problem is to find a transformation $\mathcal{T} : \mathbb{R}^D \rightarrow \mathbb{R}^D$ that maps source points to target points.

1) *Probabilistic Model*: The CPD algorithm considers the registration of the two point sets as a probability density estimation problem, where the transformed points in \mathbf{X} are the centroids of a Gaussian mixture model and the points in \mathbf{Y} as the data points generated by this model. An overview of the probabilistic model is given in Figure 2.

The first N components of the mixture model are Gaussian on the deformed source points, with mean $\tilde{\mathbf{x}}_i = \mathcal{T}(\mathbf{x}_i)$ and

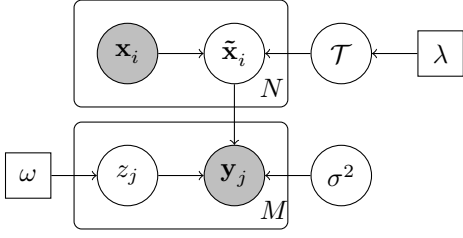


Fig. 2: Graphical model describing the generation of target points \mathbf{y}_j from a Gaussian mixture based on the source points \mathbf{x}_i . Each Gaussian has mean $\tilde{\mathbf{x}}_i = \mathcal{T}(\mathbf{x}_i)$ and variance σ^2 , $z_j = i$ indicates the component i from which the point \mathbf{y}_j is generated, λ controls the regularization of the transformation, and ω is the outlier probability.

variance σ^2 . The transformation function \mathcal{T} is the warp that we are trying to recover. An additional uniform component $N + 1$ is modeled to explain the generation of noisy and outlier points.

The prior probability of a point \mathbf{y}_j being generated from the uniform component is $P(z_j = N + 1) = \omega$, where z_j indicates a component of the mixture and ω is a parameter chosen as the outlier ratio. The prior probability of a point being generated from one of the Gaussian components is assumed to be uniform, $P(z_j = i) = (1 - \omega) \frac{1}{N}$. To lighten the notation, we denote z_{ij} to be a shorthand for $z_j = i$.

2) *Registration through Maximum a Posteriori Estimation*: The transformation function \mathcal{T} is estimated by maximizing the log-likelihood of the data, which is

$$\log P(\mathbf{Y}|\mathbf{X}, \mathcal{T}, \sigma^2) = \sum_{j=1}^M \log \sum_{i=1}^{N+1} P(z_{ij}) P(\mathbf{y}_j | z_{ij}, \mathbf{x}_i, \mathcal{T}, \sigma^2). \quad (1)$$

We can optimize this objective using the expectation-maximization (EM) algorithm [22]. Treating the component indicators z_j as latent variables, the EM framework defines the auxiliary function,

$$\mathcal{L}(q, \mathcal{T}, \sigma^2) = \sum_{j=1}^M \sum_{i=1}^{N+1} q(z_{ij} | \mathbf{y}_j) \log \frac{P(z_{ij}) P(\mathbf{y}_j | z_{ij}, \mathcal{T}, \sigma^2)}{q(z_{ij} | \mathbf{y}_j)}, \quad (2)$$

for an averaging distribution $q(z_j | \mathbf{y}_j)$. The EM algorithm is a coordinate ascent algorithm on the auxiliary function $\mathcal{L}(q, \mathcal{T}, \sigma^2)$, which is bound on the log-likelihood [22].

In the expectation step (E-step), we fix the latest transformation \mathcal{T} and variance σ^2 estimates and maximize the auxiliary function with respect to the averaging distribution q and obtain a new optimum that we denote as p_{ij} :

$$p_{ij} = \arg \max_q \mathcal{L}(q, \mathcal{T}, \sigma^2) = \frac{e^{-\|\mathbf{y}_j - \mathcal{T}(\mathbf{x}_i)\|^2 / 2\sigma^2}}{\sum_{i'=1}^N e^{-\|\mathbf{y}_j - \mathcal{T}(\mathbf{x}_{i'})\|^2 / 2\sigma^2} + \gamma}, \quad (3)$$

where $\gamma = (2\pi\sigma^2)^{D/2} \frac{\omega}{1-\omega} \frac{1}{M}$.

In the maximization step (M-step), we fix the latest distribution estimate p_{ij} and maximize the auxiliary function with respect to \mathcal{T} and σ^2 and obtain a new optimum,

$$\{\mathcal{T}, \sigma^2\} = \arg \max_{\mathcal{T}, \sigma^2} \mathcal{L}(q, \mathcal{T}, \sigma^2) = \arg \min_{\mathcal{T}, \sigma^2} E(\mathcal{T}, \sigma^2),$$

where E is the energy function

$$E(\mathcal{T}, \sigma) = \frac{1}{2\sigma^2} \sum_{i=1}^N \sum_{j=1}^M p_{ij} \|\mathbf{y}_j - \mathcal{T}(\mathbf{x}_i)\|^2 + \frac{N_P D}{2} \log \sigma^2,$$

and $N_P = \sum_{i=1}^N \sum_{j=1}^M p_{ij}$.

In order to manipulate deformable objects, we typically want a non-rigid transformation. Such transformations have many degrees of freedom and require some prior distribution to recover reasonable warps. To that end, we use the prior $P(\mathcal{T}) \propto \exp(-\frac{\lambda}{2} R(\mathcal{T}))$, where λ is a parameter and R is a regularizer. To obtain a smooth transformation \mathcal{T} , we use the thin plate spline regularizer, which is discussed in the next section. Incorporating this prior requires maximizing the posterior $P(\mathcal{T}|\mathbf{Y}, \sigma^2)$ instead of the likelihood. This is equivalent to modifying the M-step update to be

$$\{\mathcal{T}, \sigma^2\} = \arg \min_{\mathcal{T}, \sigma^2} E(\mathcal{T}, \sigma^2) + \frac{\lambda}{2} R(\mathcal{T}). \quad (4)$$

C. Thin Plate Splines

As in the previous work that uses non-rigid registration for trajectory transfer [1], we use the thin plate spline (TPS) [18], [19] parametrization for the transformation \mathcal{T} , which can be obtained by regularizing the second-order derivatives of the transformation. The TPS regularizer is given by

$$\|\mathcal{T}\|_{\text{TPS}}^2 = \int d\mathbf{x} \|\mathbf{D}^2 \mathcal{T}(\mathbf{x})\|_{\text{F}}^2, \quad (5)$$

which is a measure of the curvature of \mathcal{T} and encourages the mapping to be as smooth as possible. The optimal transformation that minimizes such regularizer can be found analytically and has the form

$$\mathcal{T}(\mathbf{x}) = \sum_i \mathbf{a}_i k(\mathbf{x}_i, \mathbf{x}) + \mathbf{B}\mathbf{x} + \mathbf{c}. \quad (6)$$

This corresponds to an affine transformation defined by \mathbf{B} and \mathbf{c} , plus a weighted sum of basis functions centered around the data points, given by $k(\mathbf{x}_i, \mathbf{x}) = -\|\mathbf{x} - \mathbf{x}_i\|^2$. We follow prior work [1] and use

$$R(\mathcal{T}) = \|\mathcal{T}\|_{\text{TPS}}^2 + (\mathbf{B} - \mathbf{I})^\top \text{diag}(\mathbf{r})(\mathbf{B} - \mathbf{I}), \quad (7)$$

where the second term is weighted by $\mathbf{r} \in \mathbb{R}^D$ and regularizes the affine part to be close to the identity matrix \mathbf{I} .

IV. LEARNING FROM MULTIPLE DEMONSTRATIONS USING TRAJECTORY-AWARE REGISTRATION

While the registration method described in the previous section can be used to transfer demonstrations for a variety of deformable object manipulation tasks, as shown in prior work [1], it is not informed by the task. The demonstrated trajectory is transformed by a non-linear warping function that aligns the demonstration point cloud to the current point cloud, under the assumption that this point cloud transformation is also appropriate for transforming the trajectory. When the two point clouds correspond to the same object, this assumption is reasonable, since points at which the trajectory interacts with the object remain on the object's surface. However, in natural environments, the point clouds will

rarely correspond to exactly the same object, so registering all points correctly is impossible. The question then arises: which points are more important to align correctly?

We use multiple demonstrations by extending the probabilistic framework in the previous section to model the point clouds of all the demonstrations, as well as the generation of trajectories in the test scene. The assumption is that if all of the demonstration trajectories are transformed by the (unknown) correct warp for the test scene, then these transformed trajectories are generated from the same distribution. This assumption is very natural, since all of the demonstrations belong to the same behavior, and the only variation between them is due to the arrangement of the scene. Optimizing the posterior probability of the new model results in an intuitive registration objective. It encodes our preference for registrations that also puts all of the demonstrations into alignment.

We combine K demonstrations of the same task to obtain a single trajectory in the test scene. To obtain the nonlinear transformation for each demonstration trajectory, we jointly optimize the registration functions $\mathcal{T}^1, \dots, \mathcal{T}^K$ for all of the demonstration point clouds together with an objective term that quantifies the degree to which the warped trajectories are aligned. Note that this procedure requires multiple demonstrations to be available in order to be trajectory-aware, since minimizing the alignment of a single trajectory with itself does not yield a meaningful objective.

In this section, we first describe how to extend the CPD model to register multiple demonstrations to the test scene, and then show how the model can be augmented to consider the distribution over transformed trajectories. Our full probabilistic model is summarized in Figure 3.

A. Non-Rigid Registration from Multiple Point Clouds

In order to learn from multiple demonstrations, we register all of the demonstration point clouds to the test scene. Let $\mathbf{X}^k = [\mathbf{x}_1^k \ \dots \ \mathbf{x}_{N_k}^k]^\top$ denote the point cloud for demonstration \mathcal{D}^k . We can register each of the demonstration scenes onto the test scene by finding K transformation functions $\mathcal{T}^1, \dots, \mathcal{T}^K$ that map points from each demonstration to the test scene. This corresponds to an extended probabilistic model where the target points are still generated from a Gaussian mixture, but now the centroids are the deformed source points from multiple demonstrations, with the centroids from demonstration \mathcal{D}^k warped by \mathcal{T}^k .

The full Gaussian mixture has $N_K + 1$ components, where $N_K = \sum_k N_k$ is the total number of source points. The first N_K components are Gaussian with mean $\tilde{\mathbf{x}}_i^k = \mathcal{T}^k(\mathbf{x}_i^k)$ and variance σ^2 . As before, the last component is uniformly distributed. The prior probability $P(z_{ij}^k)$ of a component remains the same as before, where z_{ij}^k now indexes the point i within demonstration \mathcal{D}^k . The mixture model takes the form

$$P(\mathbf{y}_j | \mathbf{x}_i^k, \mathcal{T}^{1:K}, \sigma^2) = (1 - \omega) \frac{1}{N_K} \sum_k \sum_i P(\mathbf{y}_j | z_{ij}^k, \mathbf{x}_i^k, \mathcal{T}^k, \sigma^2) + \omega \frac{1}{M}. \quad (8)$$

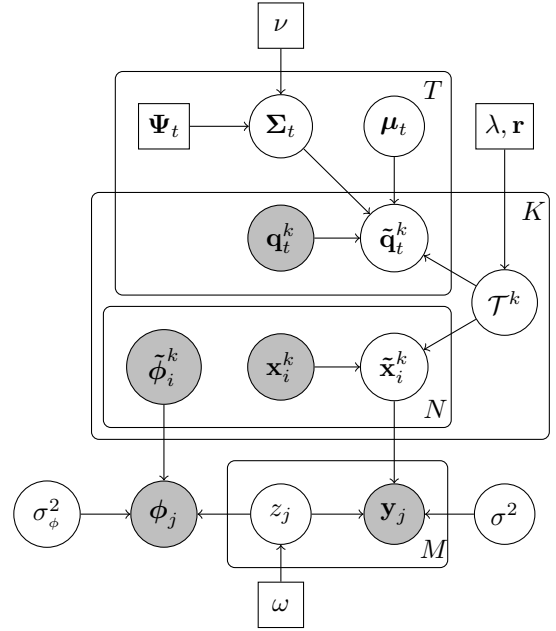


Fig. 3: Graphical model describing the generation of target points \mathbf{y}_j and features ϕ_j from a Gaussian mixture. Each Gaussian for the points has mean $\tilde{\mathbf{x}}_i^k = \mathcal{T}^k(\mathbf{x}_i^k)$ and variance σ^2 , each Gaussian for the features has mean $\tilde{\phi}_i^k$ and variance σ_ϕ^2 , $z_j = \{i, k\}$ indicates the component i from demonstration \mathcal{D}^k from which the point \mathbf{y}_j and feature ϕ_j are generated, λ and \mathbf{r} control the regularization of the transformation, and ω is the outlier probability. In addition, each transformed trajectory point $\tilde{\mathbf{q}}_t^k = \mathcal{T}^k(\mathbf{q}_t^k)$ is generated from independent Gaussians with mean μ_t and covariance Σ_t with prior parameters Ψ_t and ν .

The transformation functions $\mathcal{T}^{1:K}$ are estimated by jointly maximizing the posterior probability of the transformations, $P(\mathcal{T}^{1:K} | \mathbf{Y}, \mathbf{X}^{1:K}, \sigma^2)$. As before, we use the EM algorithm. The E-step update is similar to Equation (3), but with mixing proportions p_{ij}^k for each demonstration \mathcal{D}^k . In the M-step, we optimize with respect to the transformations $\mathcal{T}^{1:K}$ and variance σ^2 by minimizing the following objective

$$E_{\text{points}}(\mathcal{T}^1, \dots, \mathcal{T}^K, \sigma^2) = \frac{N_{\mathbf{P}} D}{2} \log \sigma^2 + \frac{1}{2\sigma^2} \sum_{k,i,j}^{K, N_k, M} p_{ij}^k \|\mathbf{y}_j - \mathcal{T}^k(\mathbf{x}_i^k)\|^2 + \frac{\lambda}{2} \sum_k R(\mathcal{T}^k), \quad (9)$$

where $N_{\mathbf{P}} = \sum_{k,i,j}^{K, N_k, M} p_{ij}^k$. In a single M-step, we first jointly optimize the transformations $\mathcal{T}^{1:K}$ while holding the variance σ^2 fixed, and then we optimize for the variance while holding the transformations fixed. This corresponds to the generalized EM algorithm [23].

We also model features $\tilde{\phi}_i^k, \phi_j \in \mathbb{R}^{D_\phi}$ of the points with a similar mixture model, where the components are Gaussian with mean $\tilde{\phi}_i^k$ and variance σ_ϕ^2 . For more details, see the Appendix.

This registration does not yet incorporate trajectory information. In the experiments, we will refer to this approach as the ablated method, and will show that incorporating trajectory-aware terms leads to a better registration.

B. Trajectory-Aware Non-Rigid Registration

While using multiple demonstrations already leads to better registration due to the improved outlier handling, it is not informed by the particular task at hand. When the scene at test time differs structurally from all of the demonstrations, simply detecting outliers may not be sufficient, since it may not even be feasible to register the inliers without excessive distortion. However, if we use the demonstrated trajectories to automatically decide which parts of the scene are relevant, we can handle significantly greater variation.

We derive a trajectory-aware registration by adding the demonstrated trajectories to the probabilistic model. In addition to the generation of points in the test scene, the model also explains the generation of trajectory points in the test scene. We parametrize a trajectory as a sequence of points in \mathbb{R}^D . These could be points on the robot's manipulator or finger as done in [13]. Let $\mathbf{Q}^k = [\mathbf{q}_1^k \dots \mathbf{q}_T^k]^\top$ be the trajectory of points of length T for each demonstration \mathcal{D}^k . These variables are observed, since they are part of the demonstration. Let $\tilde{\mathbf{Q}}^k = [\tilde{\mathbf{q}}_1^k \dots \tilde{\mathbf{q}}_T^k]^\top$ be the transformed points, with $\tilde{\mathbf{q}}_t^k = \mathcal{T}^k(\mathbf{q}_t^k)$. We model the transformed points at each time as being generated from a Gaussian distribution, with different Gaussians for each time step. The Gaussian at time step t has mean $\boldsymbol{\mu}_t$ and covariance $\boldsymbol{\Sigma}_t$. The desired trajectory for the test scene is the mean trajectory. The probability density of the transformed trajectory points is given by

$$P(\tilde{\mathbf{q}}_t^k | \mathbf{q}_t^k, \mathcal{T}^k, \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) = \frac{\exp\left(-\frac{1}{2} \|\mathcal{T}^k(\mathbf{q}_t^k) - \boldsymbol{\mu}_t\|_{\boldsymbol{\Sigma}_t^{-1}}^2\right)}{(2\pi)^{D/2} |\boldsymbol{\Sigma}_t|^{1/2}},$$

where $\|\mathbf{q} - \boldsymbol{\mu}\|_{\boldsymbol{\Sigma}^{-1}}^2 = (\mathbf{q} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{q} - \boldsymbol{\mu})$.

The parameters of these Gaussian distributions are hidden variables since they are in the transformed space and the transformations are unknown beforehand. Since there are only K data points for fitting each of these Gaussians, estimating their parameters is prone to overfitting. To overcome this, we add an inverse Wishart prior on their covariances, with scaling matrix $\boldsymbol{\Psi}_t$ and degrees of freedom ν . The prior probability of the covariance $\boldsymbol{\Sigma}_t$ is then given by

$$P(\boldsymbol{\Sigma}_t) = \frac{|\boldsymbol{\Psi}_t|^{\nu/2}}{2^{\frac{\nu D}{2}} \Gamma_D\left(\frac{\nu}{2}\right)} |\boldsymbol{\Sigma}_t|^{-\frac{\nu+D+1}{2}} \exp\left(-\frac{1}{2} \text{Tr}(\boldsymbol{\Psi}_t \boldsymbol{\Sigma}_t^{-1})\right).$$

The parameter $\boldsymbol{\Psi}_t$ reflects the relative importance of the trajectory at time step t for the registration. Intuitively, points that are closer to objects in the scene are more important, since the robot interacts with the world primarily by touching it with its grippers. Points on the trajectories that are far away from objects tend to exhibit greater random variability during the demonstrations, and are less critical to align properly in order to execute the task. Therefore, we set the parameter $\boldsymbol{\Psi}_t$ to be a diagonal matrix proportional to the average distance of the closest point, $\boldsymbol{\Psi}_t = \alpha \left(\frac{1}{K} \sum_{k=1}^K d_{kt}\right) \mathbf{I}$, where α is a proportionality constant and $d_{kt} = \min_{i \in \{1, \dots, N_K\}} \|\mathbf{q}_t^k - \mathbf{x}_i^k\|$ is the minimum

distance between the position in the trajectory at time step t and the point cloud in demonstration \mathcal{D}^k .

We estimate the parameters of these Gaussians, along with the other parameters of the model, by maximizing the posterior probability $P(\mathcal{T}^{1:K}, \boldsymbol{\Sigma}_{1:T} | \mathbf{Y}, \mathbf{X}^{1:K}, \sigma^2, \mathbf{Q}^{1:K}, \boldsymbol{\mu}_{1:T})$. The E-step remains unchanged, while the M-step now also optimizes over the trajectory covariances $\boldsymbol{\Sigma}_{1:T}$ by adding the following term to the objective

$$E_{\text{trajectories}}(\boldsymbol{\Sigma}_1, \dots, \boldsymbol{\Sigma}_T, \mathcal{T}^1, \dots, \mathcal{T}^K) = \frac{1}{2} \sum_t \sum_k \left\| \mathcal{T}^k(\mathbf{q}_t^k) - \frac{1}{K} \sum_{k'} \mathcal{T}^{k'}(\mathbf{q}_t^{k'}) \right\|_{\boldsymbol{\Sigma}_t^{-1}}^2 + \frac{\nu + K + D + 1}{2} \sum_t \log |\boldsymbol{\Sigma}_t| + \frac{1}{2} \sum_t \text{Tr}(\boldsymbol{\Psi}_t \boldsymbol{\Sigma}_t^{-1}).$$

In order to ensure that all trajectories $\mathbf{Q}^{1:K}$ are of length T , we preprocess the demonstrations by using dynamic time warping (DTW) [24] to align all of the demonstration trajectories to the first trajectory in the set.

C. Trajectory Transfer

After solving the trajectory-aware non-rigid registration optimization problem, we can compute the mean of the warped trajectory points, $\boldsymbol{\mu}_t = \frac{1}{K} \sum_k \mathcal{T}^k(\mathbf{q}_t^k)$. Using these points as the transferred trajectory, we follow prior work [13] and use trajectory optimization to find a feasible joint angle trajectory that follows the points $\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_T$. In our experiments, we take the trajectory points to be the points at the center of each finger of the robot. Multiple points per time step can be modeled by treating each of them as happening at separate time steps since, in our model, the Gaussian trajectories are independent of each other across time steps.

V. ALGORITHM SUMMARY

We have presented a probabilistic model that models the generation of point clouds, trajectories, and features for the test scene. We find a trajectory-aware non-rigid registration by optimizing the posterior probability of the graphical model in Figure 3,

$$P(\mathcal{T}^{1:K}, \boldsymbol{\Sigma}_{1:T} | \mathbf{Y}, \mathbf{X}^{1:K}, \sigma^2, \mathbf{Q}^{1:K}, \boldsymbol{\mu}_{1:T}, \boldsymbol{\Phi}, \tilde{\boldsymbol{\Phi}}^{1:K}, \sigma_\phi^2) \propto P(\mathbf{Y} | \mathbf{X}^{1:K}, \mathcal{T}^{1:K}, \sigma^2) P(\mathcal{T}^{1:K}) P(\tilde{\mathbf{Q}}^{1:K} | \mathbf{Q}^{1:K}, \mathcal{T}^{1:K}, \boldsymbol{\mu}_{1:T}, \boldsymbol{\Sigma}_{1:T}) P(\boldsymbol{\Sigma}_{1:T}) P(\boldsymbol{\Phi} | \tilde{\boldsymbol{\Phi}}^{1:K}, \sigma_\phi^2). \quad (10)$$

We optimize the posterior probability using the EM algorithm. In the E-step, we update the point correspondences between points and features in the demonstration scenes and the test scene,

$$p_{ij}^k = \frac{e^{-\|y_j - \mathcal{T}^k(\mathbf{x}_i^k)\|^2 / 2\sigma^2} e^{-\|\phi_j - \tilde{\phi}_i^k\|^2 / 2\sigma_\phi^2}}{\sum_{k', i'} e^{-\|y_j - \mathcal{T}^{k'}(\mathbf{x}_{i'}^{k'})\|^2 / 2\sigma^2} e^{-\|\phi_j - \tilde{\phi}_{i'}^{k'}\|^2 / 2\sigma_\phi^2} + \gamma}, \quad (11)$$

where $\gamma = (2\pi\sigma^2)^{D/2} (2\pi\sigma_\phi^2)^{D_\phi/2} \frac{\omega}{1-\omega} \frac{1}{M^2}$.

In the M-step, we minimize the following objective function,

$$\begin{aligned}
 & \left. \begin{aligned}
 & \frac{1}{2\sigma^2} \sum_{k,i,j}^{K,N_k,M} p_{ij}^k \|\mathbf{y}_j - \mathcal{T}^k(\mathbf{x}_i^k)\|^2 \\
 & + \frac{N_{PD}}{2} \log \sigma^2 + \frac{\lambda}{2} \sum_k^K R(\mathcal{T}^k)
 \end{aligned} \right\} E_{\text{points}} \\
 & \left. \begin{aligned}
 & + \frac{1}{2} \sum_{t,k}^{T,K} \left\| \mathcal{T}^k(\mathbf{q}_t^k) - \frac{1}{K} \sum_{k'}^K \mathcal{T}^{k'}(\mathbf{q}_t^{k'}) \right\|_{\Sigma_t^{-1}}^2 \\
 & + \frac{\nu+K+D+1}{2} \sum_t^T \log |\Sigma_t| + \frac{1}{2} \sum_t^T \text{Tr}(\Psi_t \Sigma_t^{-1})
 \end{aligned} \right\} E_{\text{trajectories}} \\
 & \left. + \frac{1}{2\sigma_\phi^2} \sum_{k,i,j}^{K,N_k,M} p_{ij}^k \|\phi_j - \tilde{\phi}_i^k\|^2 + \frac{N_{PD\phi}}{2} \log \sigma_\phi^2 \right\} E_{\text{features}}
 \end{aligned}$$

The first and last term minimizes the residual of the target points and features, and the warped source points and features from all the demonstrations, weighted by their correspondences. To enforce the smoothness of the transformation functions, we penalize each of them with the regularizer of Equation (7). The second term minimizes the weighted sum of the spatial variances of the warped trajectory points across demonstrations. In the special case that each of the trajectory covariances is fixed to be the diagonal matrix $\Sigma_t = \frac{K}{w_t} \mathbf{I}$, with weights w_t , this objective term is equivalent to a weighted sum of the trace of the empirical covariances of the warped trajectories.

VI. EXPERIMENTAL RESULTS

In this section, we present an experimental evaluation of our trajectory-aware non-rigid registration approach for learning from demonstrations on a PR2 robot. We compare our method against TPS-RPM with a single demonstration as in previous work [1], CPD with a single demonstration and the ablated method of Section IV-A that registers multiple demonstrations without using any trajectory information.

The demonstrations were obtained through kinesthetic teaching, and the green table was removed from the point clouds by using a color filter. The registration features consisted of RGB color. We used the same hyperparameters in all experiments: the outlier ratio was $\omega = 0.8$, the regularization parameters were $\lambda = 100$ and $\mathbf{r} = [10, 10, 10]$, the feature variance initialization was $\sigma_\phi^2 = 0.1$, and proportionality constant for the trajectory covariance prior was $\alpha = 10$.

A. Pick and Place from a Box with a PR2

We evaluated our approach on two pick and place tasks, which consisted of two different behaviors demonstrated for very similar scenes. The two tasks are shown in Figure 4. In the first task, shown in Figure 4a, the goal was to pick up the item in the top right corner of the box, regardless of its shape or color. In the second task, shown in Figure 4b, the goal was to always grasp the white item, regardless of its position. The last two demonstrations for both tasks were the same, due to the placement of the white object. The test

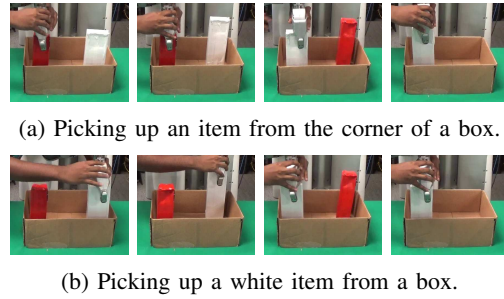


Fig. 4: Demonstrations of picking up an item from the box and placing it on the table.

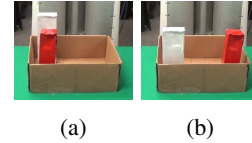


Fig. 5: Test scenes for the box tasks, (a) one with the red item in the corner and (b) the other one with the white item in the corner.

scene is shown in Figure 5, and comparisons on this scene are presented in Table I.

In the first task, all four methods are able to grab the item in the corner of the box. Without any trajectory information, the single-demonstration methods and the ablated method register as many points as possible, which in this case are the points on the box. This leads to a good registration of the box and causes the trajectories to grasp the object in the corner. In the second task, TPS-RPM, CPD and the ablated method all fail to grasp the white item. The single-demonstration methods do not capture information from multiple demonstrations and instead just take the demonstration closest to the test scene and warp its trajectory. When none of the demonstration scenes are similar to the test scene, the registration is poor and the transferred trajectory is ineffective. The ablated method registers the box at the expense of the target object, producing trajectories that reach for different parts of the box. Our trajectory-aware approach is able to capture the critical aspect of the task: the only thing that matters is the target object, since putting this object into correspondence aligns all demonstrated trajectories.

Although the prior methods are successful on one of the two tasks, our approach succeeds on both, since it is correctly able to infer which part of the scene is the most important to the task at hand by using the trajectory alignment term.

Method	Pick item at the corner		Pick white item	
	5a	5b	5a	5b
TPS-RPM	yes	yes	no	no
CPD	yes	yes	no	no
Ablated	close	yes	no	no
Trajectory-aware	yes	yes	close	yes

TABLE I: Successes of grasping the desired item, which is the item at the corner of the box for the first task and the white item for the second task. The successes for each of the two test scenes of Figure 5 are reported for each task and registration method. We report *close* when the gripper goes to the right location but misgrasps.

B. Towel Folding with a PR2

We also evaluate our approach on two towel folding tasks. The setups of these tasks are shown in Figure 6. Each of the tasks requires folding the towel, but the correct position of the fold differs between the tasks. In the first task, shown in Figure 6a, the robot must grasp the right edge of the towel and fold it to a fixed amount (three fourths of its length). In the second task, shown in Figure 6b, the robot must instead fold the towel such that the edge aligns with the white stripe. We recorded demonstrations with the stripe at different positions, as shown in Figure 6, but the same set of stripe positions was used for both tasks, so that only the motion differed. The second demonstration for both tasks was identical, since the stripe was three fourths of the way along the towel. In order to succeed at this task, the robot must associate the motion with the right cue—either the edge of the towel, or the stripe. The test scene is shown in Figure 7.

In the first task, all three methods come close to folding the towel to three fourths of its length. The single-demonstration methods succeed because the scene of the closest single demonstration matches its whole towel well with the target towel. This results in a warped trajectory leading to the correct location at three fourths the length of the towel. For the ablated method, although the stripes weren’t registered across the demonstrations, the whole towel matched the target towel, and the average trajectory brought the fold to three fourths of the length. With our method, the registration also aligns the whole towel, since the trajectories always travel the same distance and ignore the stripe.

In the second task, our method consistently placed the edge of the towel closest to the stripe. A quantitative evaluation of the final distance of the edge to the stripe is given in Table II. We can see that both of the single-demonstration methods missed the stripe, since the towel itself contains many more points, and the algorithm is not aware of the special importance of the stripe. The ablated method also did not correctly register the stripe, as shown in the point cloud renderings. Since our method minimized the variance of the trajectories, it was able to correctly pick out the stripe as the relevant cue in the scene. The registrations are shown in Table III.

These experiments show that the trajectory-aware registration method is able to correctly identify the relevant parts of the scene, while both the prior methods and the ablated method naïvely attempt to register the dissimilar scenes to one another without a careful consideration for the task.

VII. DISCUSSION AND FUTURE WORK

We presented a trajectory transfer method based on trajectory-aware non-rigid registration. Our approach transfers multiple demonstrated trajectories to a new scene by finding a non-linear warp that, along with aligning the point clouds, also aligns all of the demonstrated trajectories. By finding a transformation that minimizes the variance of points along the demonstrated trajectories, our method implicitly reduces the impact of irrelevant distractor points, thus improving generalization and robustness when compared to



(a) Folding to three fourths of the towel regardless of the placement of the white stripe.



(b) Folding towards the center of the white stripe.

Fig. 6: Demonstrations of folding a towel where the white stripe on the towel is sometimes used as a reference. The stripes were placed at 1/2, 3/4th and the end of the towel.



(a) 5/8th (b) 7/8th (c) End

Fig. 7: Test scenes for the towel tasks, with the stripe at 5/8th, 7/8th, and the end of the towel.

standard registration methods, which are not trajectory-aware and do not effectively exploit multiple demonstrations.

Our experiments demonstrate that our approach can perform complex tasks that require associating cues in the environment with parts of the demonstration. Such cues include the corner of the box and the top of the white object, as well as the stripe in the towel folding task. The results show that our trajectory-aware method achieves better generalization by using task-appropriate cues in the registration.

Although our method is trajectory-aware, we use a relatively simple geometric measure of trajectory agreement, based on the variance of the points along the multiple trajectories. While this approach is simple and effective in our experiments, it is not aware of the goals of the task, and does not make any attempt to determine which parts of each trajectory are more or less important. For instance, a demonstration that involves grasping might require much more precise positioning of the gripper during the grasp than during the rest of the motion. Our weighting scheme encodes a hand-specified notion of importance based on the distance between the objects in the point cloud, but this weighting is not adapted to the data. In future work, this kind of information could be recovered automatically by combining

Method	Fold to three fourths				Fold to stripe			
	7a	7b	7c	RMS	7a	7b	7c	RMS
TPS-RPM	3.8	1.3	1.3	2.4	22.9	4.4	3.3	13.6
CPD	3.8	0.6	1.3	2.3	31.1	5.1	34.3	26.9
Ablated	0.6	1.5	5.7	3.4	10.8	5.8	7.6	8.3
Trajectory-aware	0.6	5.8	7.6	5.6	6.4	0.6	1.3	3.8

TABLE II: Distance errors (in cm) of the towel edge and the desired positions, which is three fourths of the towel for the first task and the center of the white stripe for the second task. The errors for each of the three test scenes of Figure 7 and the root mean square (RMS) of them are reported for each task and registration method.


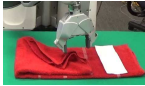






Method	TPS-RPM	CPD	Ablated	Trajectory-aware
Fold to three fourths				
Fold to stripe				

TABLE III: Results of towel folding with a PR2. Each row corresponds to a different task and the column compares our methods against using a single-demonstration with TPS-RPM and CPD, and using an ablated version of our method. The images show a snapshot of the execution. For videos, see <http://rll.berkeley.edu/iros20151fmd/>. The renderings visualize the test point cloud, the transferred gripper trajectories as dashed lines, and the mean of them as a solid line.

our approach with goal learning techniques, such as inverse reinforcement learning and inverse optimal control [25], [26], [27], [28]. An exciting avenue for future work would be to explore this direction and develop a non-rigid registration method that is not only trajectory-aware, but also goal-aware.

Acknowledgements This research was funded in part by the AFOSR through a Young Investigator Program award, by the Army Research Office through the MAST program, by the NSF NRI program under award #1227536, and by Darpa under Award #N66001-15-2-4047. Alex Lee was also supported by an NSF Fellowship.

REFERENCES

- [1] J. Schulman, J. Ho, C. Lee, and P. Abbeel, "Learning from Demonstrations through the Use of Non-Rigid Registration," in *Proceedings of the 16th International Symposium on Robotics Research (ISRR)*, 2013.
- [2] A. Lee, H. Lu, A. Gupta, S. Levine, and P. Abbeel, "Learning force-based manipulation of deformable objects from multiple demonstrations," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2015.
- [3] A. Lee, M. Goldstein, S. Barratt, and P. Abbeel, "A non-rigid point and normal registration algorithm with applications to learning from demonstrations," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2015.
- [4] J. Schulman, A. Gupta, S. Venkatesan, M. Tayson-Frederick, and P. Abbeel, "A Case Study of Trajectory Transfer Through Non-Rigid Registration for a Simplified Suturing Scenario," in *Proceedings of the 26th IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.
- [5] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Secaucus, NJ, USA: Springer, 2008, pp. 1371–1394.
- [6] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A Survey of Robot Learning from Demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, May 2009.
- [7] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in cognitive sciences*, vol. 3, no. 6, pp. 233–242, 1999.
- [8] S. Calinon, F. Guenter, and A. Billard, "On Learning, Representing, and Generalizing a Task in a Humanoid Robot," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 2, pp. 286–298, April 2007.
- [9] S. Calinon, F. D'haluin, D. Caldwell, and A. Billard, "Handling of multiple constraints and motion alternatives in a robot programming by demonstration framework," in *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, Dec 2009, pp. 582–588.
- [10] U. Hillenbrand and M. Roa, "Transferring functional grasps through contact warping and local replanning," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct 2012, pp. 2963–2970.
- [11] H. Ben Amor, O. Kroemer, U. Hillenbrand, G. Neumann, and J. Peters, "Generalization of human grasping for multi-fingered robot hands," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct 2012, pp. 2043–2050.
- [12] J. Stuckler and S. Behnke, "Efficient deformable registration of multi-resolution surfel maps for object manipulation skill transfer," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2014, pp. 994–1001.
- [13] A. X. Lee, S. H. Huang, D. Hadfield-Menell, E. Tzeng, and P. Abbeel, "Unifying scene registration and trajectory optimization for learning from demonstrations with application to manipulation of deformable objects," in *Proceedings of the 27th IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014.
- [14] A. Coates, P. Abbeel, and A. Y. Ng, "Learning for control from multiple demonstrations," in *International Conference on Machine Learning (ICML)*, 2008.
- [15] B. Argall, B. Browning, and M. Veloso, "Automatic weight learning for multiple data sources when learning from demonstration," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009.
- [16] T. Matsubara, S.-H. Hyon, and J. Morimoto, "Learning parametric dynamic movement primitives from multiple demonstrations," in *Proceedings of the 17th International Conference on Neural Information Processing: Theory and Algorithms - Volume Part I*, 2010, pp. 347–354.
- [17] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 12, pp. 2262–2275, Dec 2010.
- [18] G. Wahba, *Spline Models for Observational Data*. Philadelphia: Society for Industrial and Applied Mathematics, 1990.
- [19] F. L. Bookstein, "Principal warps: thin-plate splines and the decomposition of deformations," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 11, no. 6, pp. 567–585, Jun 1989.
- [20] H. Chui and A. Rangarajan, "A new point matching algorithm for non-rigid registration," *Computer Vision and Image Understanding*, vol. 89, no. 2-3, pp. 114–141, 2003.
- [21] J. D. Schulman, J. Ho, A. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization," in *Proceedings of Robotics: Science and Systems (RSS)*, 2013.
- [22] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.
- [23] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society, Series B*, vol. 39, no. 1, pp. 1–38, 1977.
- [24] H. Sakoe, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, pp. 43–49, 1978.
- [25] P. Abbeel and A. Ng, "Apprenticeship learning via inverse reinforcement learning," in *ICML*, 2004.
- [26] S. Levine and V. Koltun, "Continuous inverse optimal control with locally optimal examples," in *International Conference on Machine Learning (ICML)*, 2012.
- [27] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich, "Maximum margin planning," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 729–736.
- [28] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *AAAI*, 2008, pp. 1433–1438.

APPENDIX I
INCORPORATING FEATURES IN THE NON-RIGID
REGISTRATION

The registration works well when the relevant part of the object looks distinctive from the rest of it. This is because the trajectory-aware registration uses the trajectories to align the demonstration point clouds and thus learn the relevant part of the object for the task. However, if that part is indistinguishable, then the algorithm has no way of identifying the relevant part of the test object. This could happen if the distinction is in visual appearance but not in the spatial points.

For this reason, we also explicitly model visual features in our probabilistic model. Let the features of the source point set be $\tilde{\Phi}^k = [\tilde{\phi}_1^k \dots \tilde{\phi}_{N_K}^k]^\top$ and the features of the target point set be $\Phi = [\phi_1 \dots \phi_M]^\top$, with features $\tilde{\phi}_i^k, \phi_j \in \mathbb{R}^{D_\phi}$, where D_ϕ is the dimension of the features. The feature $\tilde{\phi}_i^k$ corresponds to point \mathbf{x}_i^k and the feature ϕ_j corresponds to point \mathbf{y}_j . Unlike the points, we assume that the features are invariant to the transformation function. In our experiments, we use the RGB color of the point as the feature for that point. That is, the feature is a 3-dimensional vector with one color channel per entry and the domain of each channel is between 0 and 1. The probabilistic model resembles the mixture in Section IV-A but now the components are Gaussian with mean $\tilde{\phi}_i^k$ and variance σ_ϕ^2 ,

$$P(\phi_j | \tilde{\phi}_i^k, \sigma_\phi^2) = (1 - \omega) \frac{1}{N_K} \sum_k \sum_i P(\phi_j | z_{ij}^k, \tilde{\phi}_i^k, \sigma_\phi^2) + \omega \frac{1}{M}. \quad (12)$$

We estimate the parameters by maximizing the posterior $P(\mathcal{T}^{1:K}, \Sigma_{1:T} | \mathbf{Y}, \mathbf{X}^{1:K}, \sigma^2, \mathbf{Q}^{1:K}, \boldsymbol{\mu}_{1:T}, \Phi, \tilde{\Phi}^{1:K}, \sigma_\phi^2)$. The M-step now also optimizes over the variance σ_ϕ^2 and contributes the following term to the objective

$$E_{\text{features}}(\sigma_\phi) = \frac{1}{2\sigma_\phi^2} \sum_{k,i,j}^{K, N_k, M} p_{ij}^k \left\| \phi_j - \tilde{\phi}_i^k \right\|^2 + \frac{N_P D_\phi}{2} \log \sigma_\phi^2. \quad (13)$$

APPENDIX II
COHERENT POINT DRIFT WITH THIN PLATE SPLINES IN
CLOSED-FORM

The thin plate spline form of Equation (6) uses a weighted sum of basis functions centered around the data points. In this section we consider the general case in which the basis functions are centered around any points. These points are called control points, which we denote as $\hat{\mathbf{X}} = [\hat{\mathbf{x}}_1 \dots \hat{\mathbf{x}}_{\hat{N}}]^\top$. In our experiments, the control points that we use are points of a downsampled version of the source point cloud. This reduces the number of variables to optimize, which gives us computation speedup without any noticeable effect on the expressiveness of the thin plate spline transformation. The

closed form of the thin plate spline is then [18], [19]

$$\mathcal{T}(\mathbf{x}) = \sum_{i=1}^{\hat{N}} \mathbf{a}_i k(\hat{\mathbf{x}}_i, \mathbf{x}) + \mathbf{B}\mathbf{x} + \mathbf{c}, \quad (14)$$

where the basis function is given by $k(\hat{\mathbf{x}}_i, \mathbf{x}) = -\|\mathbf{x} - \hat{\mathbf{x}}_i\|^2$ and the weights $\mathbf{A} = [\mathbf{a}_1 \dots \mathbf{a}_{\hat{N}}]^\top$ are constrained by

$$\mathbf{A}^\top \begin{bmatrix} \hat{\mathbf{X}} & \mathbf{1}_{\hat{N} \times 1} \end{bmatrix} = \mathbf{0}_{D \times (D+1)}. \quad (15)$$

Let $\mathbf{U}_{\hat{\mathbf{X}}}^{\hat{\mathbf{X}}}$ be a matrix of basis functions between the control points $\hat{\mathbf{X}}$ and points \mathbf{X} ,

$$\mathbf{U}_{\hat{\mathbf{X}}}^{\hat{\mathbf{X}}} = \begin{bmatrix} k(\hat{\mathbf{x}}_1, \mathbf{x}_1) & \dots & k(\hat{\mathbf{x}}_{\hat{N}}, \mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ k(\hat{\mathbf{x}}_1, \mathbf{x}_N) & \dots & k(\hat{\mathbf{x}}_{\hat{N}}, \mathbf{x}_N) \end{bmatrix} \quad (16)$$

We can write the transformation of points \mathbf{X} in matrix form as

$$\mathcal{T}(\mathbf{X}) = \mathbf{U}_{\hat{\mathbf{X}}}^{\hat{\mathbf{X}}} \mathbf{A} + \mathbf{X} \mathbf{B}^\top + \mathbf{1} \mathbf{c}^\top = \mathbf{G}_{\hat{\mathbf{X}}}^{\hat{\mathbf{X}}} \Theta, \quad \mathbf{G}_{\hat{\mathbf{X}}}^{\hat{\mathbf{X}}} = \begin{bmatrix} \mathbf{U}_{\hat{\mathbf{X}}}^{\hat{\mathbf{X}}} & \mathbf{X} & \mathbf{1}_{N \times 1} \end{bmatrix}, \quad \Theta = \begin{bmatrix} \mathbf{A} \\ \mathbf{B}^\top \\ \mathbf{c}^\top \end{bmatrix}.$$

The application of the transformation \mathcal{T} to a matrix \mathbf{X} is defined to be the element-wise application of the transformation to the vectors in \mathbf{X} .

The thin plate spline regularizer of Equation (5) can be expressed in closed form as

$$\|\mathcal{T}\|_{\text{TPS}}^2 = \text{Tr} \left(\mathbf{A}^\top \mathbf{U}_{\hat{\mathbf{X}}}^{\hat{\mathbf{X}}} \mathbf{A} \right). \quad (17)$$

The parameters Θ of the thin plate spline are constrained by Equation (15). Alternatively, we can use unconstrained parameters \mathbf{Z} by using the change of variables,

$$\Theta = \mathbf{N}^{\hat{\mathbf{X}}} \mathbf{Z}, \quad (18)$$

$$\mathbf{N}^{\hat{\mathbf{X}}} = \begin{bmatrix} \mathbf{I}_{(D+1) \times (D+1)} & \mathbf{0} \\ \mathbf{0} & \text{null} \left(\begin{bmatrix} \hat{\mathbf{X}} & \mathbf{1}_{\hat{N} \times 1} \end{bmatrix}^\top \right) \end{bmatrix}. \quad (19)$$

The optimization problem of Equation (4) with respect to the transformation \mathcal{T} is equivalent to

$$\min_{\mathbf{Z}} \frac{1}{2} \text{Tr}(\mathbf{Z}^\top \mathbf{H} \mathbf{Z}) - \text{Tr}(\mathbf{Z}^\top \mathbf{f}), \quad (20)$$

where

$$\mathbf{H} = \frac{1}{\sigma^2} (\mathbf{G}_{\hat{\mathbf{X}}}^{\hat{\mathbf{X}}} \mathbf{N}^{\hat{\mathbf{X}}})^\top \mathbf{d}(\mathbf{P} \mathbf{1}) (\mathbf{G}_{\hat{\mathbf{X}}}^{\hat{\mathbf{X}}} \mathbf{N}^{\hat{\mathbf{X}}}) + \lambda (\mathbf{N}^{\hat{\mathbf{X}}})^\top \mathbf{S}^{\hat{\mathbf{X}}} \mathbf{N}^{\hat{\mathbf{X}}},$$

$$\mathbf{f} = \frac{1}{\sigma^2} (\mathbf{G}_{\hat{\mathbf{X}}}^{\hat{\mathbf{X}}} \mathbf{N}^{\hat{\mathbf{X}}})^\top \mathbf{P} \mathbf{Y} + \lambda (\mathbf{N}^{\hat{\mathbf{X}}})^\top \mathbf{s}^{\hat{\mathbf{X}}},$$

$$\mathbf{S}^{\hat{\mathbf{X}}} = \begin{bmatrix} \mathbf{U}_{\hat{\mathbf{X}}}^{\hat{\mathbf{X}}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{d}(\mathbf{r}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad \mathbf{s}^{\hat{\mathbf{X}}} = \begin{bmatrix} \mathbf{0}_{\hat{N} \times D} \\ \mathbf{d}(\mathbf{r}) \\ \mathbf{0}_{1 \times D} \end{bmatrix}.$$

The operator $\mathbf{d}(\cdot)$ constructs a diagonal matrix from the elements of a vector, or a block diagonal matrix from a sequence of matrices.

The optimal thin plate spline is given by the parameters $\mathbf{Z} = \mathbf{H}^{-1} \mathbf{f}$, or $\Theta = \mathbf{N}^{\hat{\mathbf{X}}} \mathbf{H}^{-1} \mathbf{f}$.

APPENDIX III

 TRAJECTORY-AWARE NON-RIGID REGISTRATION IN
CLOSED-FORM

As summarized in Section V, in the E-step we update the correspondences as given by Equation (11), and in the M-step we minimize the objective

$$E_{\text{points}}(\mathcal{T}^{1:K}, \sigma^2) + E_{\text{trajectories}}(\Sigma_{1:T}, \mathcal{T}^{1:K}) + E_{\text{features}}(\sigma_\phi)$$

with respect to the transformations $\mathcal{T}^{1:K}$, the point and feature variances σ^2 and σ_ϕ^2 , and the trajectory covariances $\Sigma_{1:T}$. The variables of the objective above cannot be optimized jointly in closed-form. However, each of those sets of variables can be optimized in closed-form while holding the other ones fixed. In that case, the optima are given by

$$\begin{aligned} \mathcal{T}^{1:K} = \arg \min_{\mathcal{T}^{1:K}} \frac{1}{2\sigma^2} \sum_{k,i,j}^{K,N_k,M} \|\mathbf{y}_j - \mathcal{T}^k(\mathbf{x}_i^k)\|^2 \\ + \frac{\lambda}{2} \sum_k^K R(\mathcal{T}^k) + \frac{1}{2} \sum_{t,k}^{T,K} \left\| \mathcal{T}^k(\mathbf{q}_t^k) - \frac{1}{K} \sum_{k'}^K \mathcal{T}^{k'}(\mathbf{q}_t^{k'}) \right\|_{\Sigma_t^{-1}}^2 \end{aligned} \quad (21)$$

$$\sigma^2 = \frac{1}{N_{\mathcal{P}D}} \sum_{k,i,j}^{K,N_k,M} p_{ij}^k \|\mathbf{y}_j - \mathcal{T}^k(\mathbf{x}_i^k)\|^2 \quad (22)$$

$$\sigma_\phi^2 = \frac{1}{N_{\mathcal{P}D_\phi}} \sum_{k,i,j}^{K,N_k,M} p_{ij}^k \|\phi_j - \tilde{\phi}_i^k\|^2 \quad (23)$$

$$\Sigma_t = \frac{\Psi_t + K \hat{\Sigma}_t}{\nu + K + D + 1}, \quad (24)$$

where $N_{\mathcal{P}} = \sum_{k,i,j}^{K,N_k,M} p_{ij}^k$ and $\hat{\Sigma}_t$ is the empirical covariance of the transformed trajectory points,

$$\hat{\Sigma}_t = \frac{1}{K} \sum_k^K \left(\tilde{\mathbf{q}}_t^k - \frac{1}{K} \sum_{k'}^K \tilde{\mathbf{q}}_t^{k'} \right) \left(\tilde{\mathbf{q}}_t^k - \frac{1}{K} \sum_{k'}^K \tilde{\mathbf{q}}_t^{k'} \right)^\top,$$

with $\tilde{\mathbf{q}}_t^k = \mathcal{T}^k(\mathbf{q}_t^k)$ being a function of the transformation \mathcal{T}^k .

Denote $\mathbf{Z} = [(\mathbf{Z}^1)^\top \dots (\mathbf{Z}^K)^\top]^\top$ to be the concatenation of the parameters $\mathbf{Z}^1, \dots, \mathbf{Z}^K$ for the respective transformations $\mathcal{T}^1, \dots, \mathcal{T}^K$. The optimization of Equation (21) is a quadratic function of the parameters \mathbf{Z} so it can be solved in closed-form. The optimization has the form of Equation (20), but now the matrices \mathbf{H} and \mathbf{f} are block-diagonal versions of the ones in the previous section.

We constrain $\Sigma_t = \sigma_{\mathbf{q}_t}^2 \mathbf{I}$ to be a diagonal matrix with the same variance along the diagonal. In this case, the update of Equation (24) can further be simplified to

$$\sigma_{\mathbf{q}_t}^2 = \frac{\text{Tr}(\Psi_t + K \hat{\Sigma}_t)}{D(\nu + K + D + 1)}. \quad (25)$$

The overall algorithm is summarized in Algorithm 1 and it includes the closed-form updates for the M-step. We set $\nu = K + 2$ in our implementation, which is a standard default choice.

Algorithm 1 Trajectory-Aware Non-Rigid Registration

```

1: procedure TRAJAWAREREG( $\mathbf{X}^{1:K}, \mathbf{Y}, \tilde{\Phi}^{1:K}, \Phi, \mathbf{Q}^{1:K},$ 
    $\omega, \lambda, \mathbf{r}, \Psi_{1:T}, \nu$ )
2:                                     ▷ Precomputation
3:    $\mathbf{N} \leftarrow \text{d}(\mathbf{N}^{\hat{\mathbf{X}}^1}, \dots, \mathbf{N}^{\hat{\mathbf{X}}^K})$ 
4:    $\mathbf{GN} \leftarrow \text{d}(\mathbf{G}_{\hat{\mathbf{X}}^1}^{\hat{\mathbf{X}}^1} \mathbf{N}^{\hat{\mathbf{X}}^1}, \dots, \mathbf{G}_{\hat{\mathbf{X}}^K}^{\hat{\mathbf{X}}^K} \mathbf{N}^{\hat{\mathbf{X}}^K})$ 
5:    $\mathbf{S} \leftarrow \text{d}(\mathbf{S}^{\hat{\mathbf{X}}^1}, \dots, \mathbf{S}^{\hat{\mathbf{X}}^K})$ 
6:    $\mathbf{s} \leftarrow [(\mathbf{s}^{\hat{\mathbf{X}}^1})^\top \dots (\mathbf{s}^{\hat{\mathbf{X}}^K})^\top]^\top$ 
7:   for all  $t \in \{1, \dots, T\}$  do
8:      $\mathbf{L}_t \leftarrow \text{d}(\mathbf{G}_{\mathbf{q}_t^1}^{\hat{\mathbf{X}}^1} \mathbf{N}^{\hat{\mathbf{X}}^1}, \dots, \mathbf{G}_{\mathbf{q}_t^K}^{\hat{\mathbf{X}}^K} \mathbf{N}^{\hat{\mathbf{X}}^K})$ 
9:   end for
10:                                     ▷ Initialization
11:    $\sigma^2 \leftarrow \frac{1}{DN_{KM}} \sum_{k,i,j}^{K,N_k,M} \|\mathbf{y}_j - \mathbf{x}_i^k\|^2$ 
12:    $\sigma_\phi^2 \leftarrow \frac{1}{D_\phi N_{KM}} \sum_{k,i,j}^{K,N_k,M} \|\phi_j - \tilde{\phi}_i^k\|^2$ 
13:   for all  $t \in \{1, \dots, T\}$  do
14:      $\sigma_{\mathbf{q}_t}^2 \leftarrow \frac{1}{D} \text{Tr}(\Psi_t)$ 
15:   end for
16:   repeat                                     ▷ EM loop
17:                                     ▷ E-step
18:      $\gamma \leftarrow (2\pi\sigma^2)^{D/2} (2\pi\sigma_\phi^2)^{D_\phi/2} \frac{\omega}{1-\omega} \frac{1}{M^2}$ 
19:     for all  $k \in \{1, \dots, K\}$  do
20:       for all  $i, j \in \{1, \dots, N_k\} \times \{1, \dots, M\}$  do
21:          $p_{ij}^k \leftarrow \frac{e^{-\|\mathbf{y}_j - \mathcal{T}^k(\mathbf{x}_i^k)\|^2/2\sigma^2} e^{-\|\phi_j - \tilde{\phi}_i^k\|^2/2\sigma_\phi^2}}{\sum_{k',i'}^{K,N_k} e^{-\|\mathbf{y}_j - \mathcal{T}^{k'}(\mathbf{x}_{i'}^{k'})\|^2/2\sigma^2} e^{-\|\phi_j - \tilde{\phi}_{i'}^{k'}\|^2/2\sigma_\phi^2} + \gamma}$ 
22:       end for
23:     end for
24:      $\mathbf{P} \leftarrow [(\mathbf{P}^1)^\top \dots (\mathbf{P}^K)^\top]^\top$ 
25:                                     ▷ M-step
26:      $\mathbf{H} \leftarrow \frac{1}{\sigma^2} (\mathbf{GN})^\top \text{d}(\mathbf{P}\mathbf{1})(\mathbf{GN}) + \lambda \mathbf{N}^\top \mathbf{S} \mathbf{N}$ 
        $+ \sum_{t=1}^T \frac{1}{\sigma_{\mathbf{q}_t}^2} \mathbf{L}_t^\top (\mathbf{I}_{K \times K} - \frac{1}{K} \mathbf{1}_{K \times K}) \mathbf{L}_t$ 
27:      $\mathbf{f} \leftarrow \frac{1}{\sigma^2} (\mathbf{GN})^\top \mathbf{P} \mathbf{Y} + \lambda \mathbf{N}^\top \mathbf{s}$ 
28:      $\mathbf{Z} \leftarrow \mathbf{H}^{-1} \mathbf{f}$ 
29:     Update  $\mathcal{T}^1, \dots, \mathcal{T}^K$  with parameters  $\mathbf{Z}$ 
30:      $\tilde{\mathbf{X}} \leftarrow \mathbf{GN} \mathbf{Z}$ 
31:      $N_{\mathcal{P}} \leftarrow \mathbf{1}^\top \mathbf{P} \mathbf{1}$ 
32:      $\sigma^2 \leftarrow \frac{1}{N_{\mathcal{P}D}} \left( \text{Tr}(\mathbf{Y}^\top \text{d}(\mathbf{P}^\top \mathbf{1}) \mathbf{Y}) \right.$ 
        $- 2 \text{Tr}((\mathbf{P} \mathbf{Y})^\top \tilde{\mathbf{X}})$ 
        $\left. + \text{Tr}(\tilde{\mathbf{X}}^\top \text{d}(\mathbf{P} \mathbf{1}) \tilde{\mathbf{X}}) \right)$ 
33:      $\sigma_\phi^2 \leftarrow \frac{1}{N_{\mathcal{P}D_\phi}} \left( \text{Tr}(\Phi^\top \text{d}(\mathbf{P}^\top \mathbf{1}) \Phi) \right.$ 
        $- 2 \text{Tr}((\mathbf{P} \Phi)^\top \tilde{\Phi}^{1:K})$ 
        $\left. + \text{Tr}((\tilde{\Phi}^{1:K})^\top \text{d}(\mathbf{P} \mathbf{1}) \tilde{\Phi}^{1:K}) \right)$ 
34:     for all  $t \in \{1, \dots, T\}$  do
35:        $\tilde{\mathbf{Q}}_t \leftarrow \mathbf{L}_t \mathbf{Z}$ 
36:        $\boldsymbol{\mu}_t \leftarrow \frac{1}{K} \tilde{\mathbf{Q}}_t^\top \mathbf{1}_{K \times 1}$ 
37:        $\hat{\Sigma}_t \leftarrow \frac{1}{K} \left( \tilde{\mathbf{Q}}_t - \mathbf{1} \boldsymbol{\mu}_t^\top \right)^\top \left( \tilde{\mathbf{Q}}_t - \mathbf{1} \boldsymbol{\mu}_t^\top \right)$ 
38:        $\sigma_{\mathbf{q}_t}^2 \leftarrow \frac{1}{D(\nu+K+D+1)} \text{Tr}(\Psi_t + K \hat{\Sigma}_t)$ 
39:     end for
40:   until convergence
41:   return  $\mathcal{T}^1, \dots, \mathcal{T}^K, \boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_T$ 
42: end procedure

```