Probabilistic Decomposition of Sequential Force Interaction Tasks into Movement Primitives

Simon Manschitz[1,2], Michael Gienger[2], Jens Kober[3], and Jan Peters[1,4]

***Abstract*— Learning sequential force interaction tasks from kinesthetic demonstrations is a promising approach to transfer human manipulation abilities to a robot. In this paper we propose a novel concept to decompose such demonstrations into a set of Movement Primitives (MPs). The decomposition is based on a probability distribution we call Directional Normal Distribution (DND). To capture the sequential properties of the manipulation task, we model the demonstrations with a Hidden Markov Model (HMM). Here, we employ mixtures of DNDs as the HMM's output emissions. The combination of HMMs and mixtures of DNDs allows to infer the MP's composition, i.e., its coordinate frames, control variables and target coordinates from the demonstration data. In addition, it permits to determine an appropriate number of MPs that explains the demonstrations best. We evaluate the approach on kinesthetic demonstrations of a light bulb unscrewing task. Decomposing the task leads to intuitive and meaningful MPs that reflect the natural structure of the task.**

I. Introduction

Imitation learning of robot manipulation tasks is a promising research area with many potential application domains. These range from industrial applications such as assembly to service robotics. Intuitive learning approaches can be a tool to speed up the programming process, as well as a means to realize tasks with higher complexity. This paper's contribution is a concept to improve the human-to-robot skill transfer of sequential force interaction tasks. Such tasks may involve physical contact between the robot and its environment. Examples are assembly problems where a robot may exert a force on an object such as pushing a button, or rotating a handle while pulling it. Naturally, such tasks can be decomposed into a set of elementary movements, often called movement primitives (MPs). In the remainder of the paper, we define an MP as an elementary movement that is composed of a spatially represented dynamical system with attractor behavior $\dot{\boldsymbol{x}} = f(\boldsymbol{x})$. Vector $\boldsymbol{x}$ is composed of a set of task-space control variables, which may be represented in different coordinate frames [1]. Control variables may for instance be a spatial position or orientation of an end-effector in object coordinates, or a force of the end effector in world coordinates. The dynamical system behavior $f(\boldsymbol{x})$ is assumed to have a convergent attractor behavior, making the MP eventually converge to the target coordinates of the control variables. We further assume that the composition (control variables and coordinate frames) for an MP are constant.

To avoid defining a specific MP for each subtask by hand, the learning from demonstrations paradigms suggests to learn them automatically from demonstrations of the task. Our focus is on learning from kinesthetic demonstrations. In a kinesthetic demonstration, a teacher guides the robot physically through a task, similar to a parent teaching a task to his or her child. Such demonstrations allow to record both kinematics and force sensor data in the robot's embodiment, hence circumventing the correspondence problem. Further, the force measurements are an important source of information when inferring which control variable to use, e.g., force and/or position-control. The aim of our approach is to decompose a set of kinesthetic demonstrations of a force interaction task into its generating MPs. We present a concept to infer the composition of each MP, i.e., its control variables, target coordinates and coordinate frames. In addition, we do not assume to know the number of involved MPs beforehand, but will infer them from the data.

A. Related Work

Decomposing a task based on kinesthetic demonstrations has received a lot of attention in robotics research over the past years. Many researchers concentrate on segmenting demonstrations on a trajectory level, mainly neglecting forces. Niekum et al. [2] segmented demonstrations using an auto-regressive Hidden Markov Model. In order to find a task-decomposition that does not only explain the data well but is also semantically meaningful for the task, they afterwards split up states of the resulting model to build a Finite State Machine representation based on prior knowledge. Lioutikov et al. [3] assume an initial guess for potential cut points between successive MP segments. They then try to prune out false positive cut points and find the MPs responsible for generating the data of each segment in a probabilistic way using an Expectation-Maximization (EM) algorithm. Both approaches do not consider different coordinate frames and neglect forces.

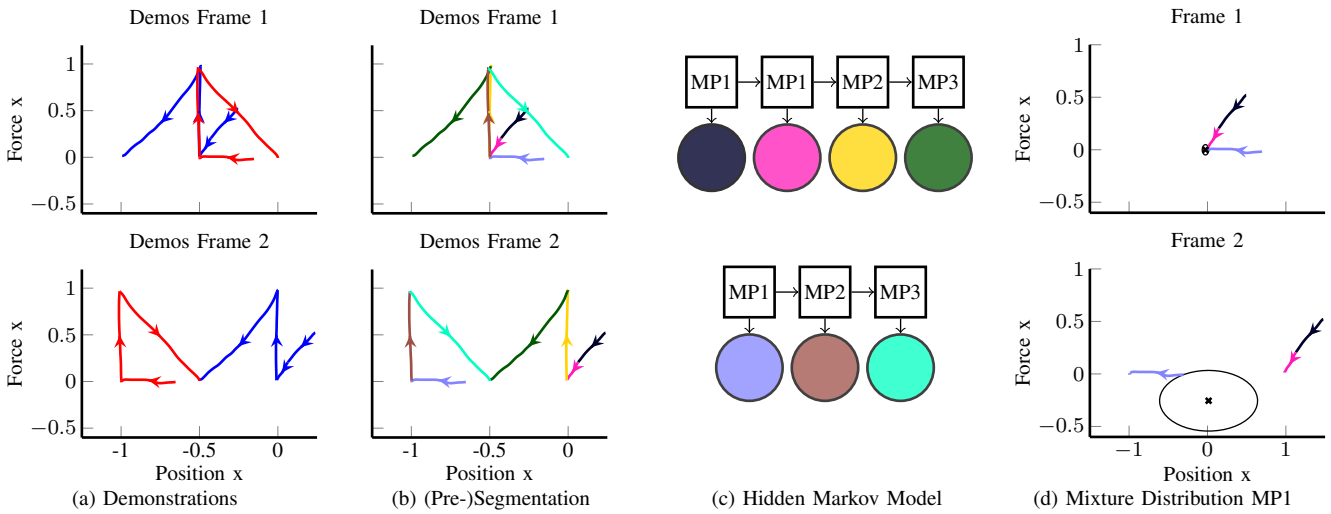
Other comparable approaches use different methods and models, but also try to find a segmentation without taking into account the measured forces. Examples for other methods are Inverse Reinforcement Learning [4], [5], Transition

[1]S. Manschitz and J. Peters are with the Institute for Intelligent Autonomous Systems, Technische Universität Darmstadt, 64289 Darmstadt, Germany, manschitz@ias.tu-darmstadt.de, mail@jan-peters.net

[2]S. Manschitz and M. Gienger are with the Honda Research Institute Europe, 63073 Offenbach, Germany, michael.gienger@honda-ri.de

[3]J. Kober is with the Delft Center for Systems and Control, Delft University of Technology, 2628 CD, Delft, The Netherlands, j.kober@tudelft.nl

[4]J. Peters is with the Max Planck Institute for Intelligent Systems, 72076 Tübingen, Germany



(a) Demonstrations (b) (Pre-)Segmentation (c) Hidden Markov Model (d) Mixture Distribution MP1

Fig. 1: Overview of our task-decomposition approach using a simple 1D toy example. Plot (a) shows two demonstrations (blue and red) in two different coordinate frames. Our approach first (pre-)segments the data by finding zero-velocity crossings and contact changes (b). The different segments are illustrated using different colors. Then, the segments are assigned to MPs using a Hidden Markov Model (HMM) (c). Here, we use mixtures of Directional Normal Distributions (DNDs) as output emissions of the HMM. DNDs are introduced in this paper and allow us to to extract the the MP parameters (coordinate frame, control variables, target) after training the HMM. An illustration of the extraction can be seen in (d) for MP1. The upper plot shows the mean (target) of the distribution after training for the first coordinate frame (the first mixture) and the lower plot shows the mean for the second coordinate frame. The ellipsoids indicate the uncertainty of the mean. As the uncertainty is much larger for the second frame, the first coordinate frame is more likely for this MP. Additionally, the force converges to zero, which is why our algorithm chooses position as control variable for this MP.

State Clustering [6], Bayesian Binning [7], Hidden Markov Models [8]–[10], or Conditional Random Fields [11], [12].

An approach that explicitly deals with coordinate frames is that of Rozo et al. [13]. The authors use a task-parametrized Gaussian mixture model (TP-GMM) as trajectory representation. A TP-GMM is a hierarchical GMM with two layers, meaning that each mixture is again a GMM. For each mixture, the nested GMM is weighting the importance of the different coordinate frames. In addition, they learn to vary the controller stiffness dynamically, so that the robot can physically interact with a human co-worker. Abu-Dakka et al. [14] adapt the dynamic movement primitives (DMP) framework so that the robot follows a desired force profile. Yet, they predefine if an MP is position or force-controlled. Steinmetz et al. [15] learn a desired position and force profile with a DMP from a single demonstration. They present a control framework that is able to transition between phases of pure impedance control and force control based on the measured external force. The approach is not able to handle multiple demonstrations or sequences of MPs.

Ureche et al. [16] extract the coordinate frames and control variables based on the variance of the data. If the variance of a variable is large within a time window for all demonstrations, they consider it to be significant for the task. The reason is that the variable changes its value and does that in a systematic way across all demonstrations. Kober et al. [17] showed that this assumption leads to an over-segmentation of the data. Therefore, they suggested to incorporate the convergence behavior of the motion as well. In order to choose the coordinate frame, they compute a scoring function that reinforces if a variable is converging to a target for a given segment. The main disadvantage of both approaches is that for computing the statistics, they need to know a priori which segments of the demonstrations represent the same MP. For their experiments this knowledge is not necessary, as the MP sequence necessary to perform the task is always the same. Therefore, they can align the data over time with Dynamic Time Warping, allowing a comparison of the data over the different demonstrations.

Many approaches (e.g., [16]–[18]) require demonstrations with identical sequential ordering of the employed MPs. Our approach does not rely on aligning the demonstrations in time and is therefore capable of decomposing demonstrations with varying sequential MP orders (e.g., unscrewing tasks). Furthermore, our approach is able to simultaneously infer the MP sequence as well as the composition of the MPs. We will show that decomposing kinesthetic demonstrations of a force interaction task with our approach leads to meaningful and intuitive MPs.

The remainder of this paper is organized as follows. We give an overview of our approach in the following Section II. In Section III, we introduce the probability distribution that is used within our task-decomposition approach. In Section IV, we evaluate our method on real data from kinesthetic demonstrations of a light bulb unscrewing task. Finally, we conclude and give a short outlook on future work in Section V.

II. Proposed Task-Decomposition Approach

Before we describe our approach in detail, we introduce the notation that will be used throughout the paper. For each time step of a demonstration i with length $N_{\mathrm{T_i}}$, we record the N_{J} joint angles $\boldsymbol{Q}^{(i)} \in \mathbb{R}^{N_{\mathrm{J}} \times N_{\mathrm{T_i}}}$, the six forces and torques from the force-torque sensor at the wrist of the robot $\boldsymbol{F}^{(i)} \in \mathbb{R}^{6 \times N_{\mathrm{T_i}}}$, and the 3D positions and 3D

orientations of all N_{O} objects in the scene $\boldsymbol{O}^{(i,j)} \in \mathbb{R}^{6\times N_{\mathrm{Ti}}}$. The joint angles also include the joints of the end-effector. Here, $\boldsymbol{O}^{(i,j)}$ is the matrix containing the positions and orientations of the jth object. Each object and the world are associated with a coordinate frame, forming a set of $N_{\mathrm{TS}} = N_{\mathrm{O}} + 1$ task-spaces. After the demonstrations, the data are projected on the task-spaces, resulting in the matrices $\boldsymbol{X}^{(i,k)} \in \mathbb{R}^{N_{\mathrm{Xk}}\times N_{\mathrm{Ti}}}$, where N_{Xk} is the dimension of the kth task-space.

The overview of our proposed task-decomposition approach can be found in Figure 1. In order to decompose the task, our approach performs three basic steps that will be explained in detail in the following subsections.

A. Segmentation

In the first step of the approach, we detect potential cuts between successive segments (see Figure 1b). A cut is defined at a point where the robot stops or if a contact event occurs (e.g., robot gets into contact or loses contact with an object). Flanagan et al. [19] showed the relevance of this definition for dexterous manipulation from the biological point of view. We detect if the robot moves or stops by finding the zero-velocity crossings (ZVCs) of the end-effector position and/or orientation in world coordinates. Additionally, a cut is added for ZVCs of the finger joint angles. Contact events are found by detecting the zero-velocity crossings of the measured forces and torques. The segmentation splits the task-space data into a sequence of segments $\boldsymbol{X}^{(i,k,l)} \in \mathbb{R}^{N_{\mathrm{Xk}}\times N_{\mathrm{Tl}}}$, where N_{Tl} is the length of the lth segment of demonstration i.

B. Assignment of Segments to MPs

In the second step, we assign each segment to an MP as illustrated in Figure 1c. Since we assume a convergent attractor behavior of the MPs, we argue that segments (sequences of data points) converging to the same target (in at least one task-space) should be assigned to the same MP. Therefore, we introduce a probability distribution we call Directional Normal Distribution (DND). Intuitively, a DND works as follows. As for a Gaussian distribution, the parameters of a DND are a mean $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$. Given at least one segment, its mean is defined as the point the segment is converging to and the covariance matrix describes the uncertainty of the mean. Thus, the mean can be interpreted as the most likely target of an MP corresponding to a given segment (or given segments). In Section III, we explain in more detail how the learning algorithm for the DNDs works. For now, it is sufficient to know that given some segments, we can estimate the target of an MP in a probabilistic way by using a DND.

For finding the most likely assignment of segments to MPs, we train a Hidden Markov Model (HMM) with mixtures of DNDs as output emissions. For the HMM, each segment corresponds to an observation and each hidden state corresponds to an MP. While the HMM takes into account the sequential properties of the demonstrations, the mixtures are used to represent the demonstrations in the different task-spaces. For estimating the number of hidden states (the number of MPs), we use the Bayesian Information Criterion (BIC), a standard model selection criterion for probabilistic models. The HMM training algorithm is initialized with a varying number of MPs and the model with the lowest resulting BIC value is selected in the end.

C. Extraction of MP Parameters

In the last step of our approach, the MP parameters are extracted from the HMM after training, as illustrated in Figure 1d. For each hidden state of the HMM, we can extract the parameters of the corresponding MP from the mixtures of DNDs. The highest mixture weight indicates the most likely coordinate frame. The MP target can be extracted from the mean of the corresponding DND. Finally, if the task-space of an MP is composed of forces and positions along the same axis, we explicitly decide whether force or position are chosen as control variable. Here, we use of a simple heuristic. Force is only chosen if the desired attractor force of the MP is higher than a threshold and the velocity mean of the segments assigned to this MP is below a threshold. Thus, if a force was measured along a certain axis and the robot was not moving in this direction, we assume the teacher wanted the robot to apply a force.

III. Directional Normal Distribution

We introduce the DND as we want to estimate the most likely target of an MP for a given a set of segments. Therefore, a DND has a mean $\boldsymbol{\mu}$ (a target) and a covariance matrix $\boldsymbol{\Sigma}$ (indicating the uncertainty of the target) as parameters. Our basic assumption is that for each data point $\boldsymbol{x} \in \mathbb{R}^{d\times 1}$ from one of the segments, the corresponding velocity vector $\boldsymbol{v} \in \mathbb{R}^{d\times 1}$ roughly points towards the target of the MP. The target is then defined as the point that is most consistent with all velocity vectors. Please note that a DND has no notion of a segment. Instead, it assigns each data point of a segment a probability. Therefore, we will omit the notion of a segment in the following and assume that we work on a set of N data points instead. The idea behind the DND and the learning algorithm that will be explained in the following sections are illustrated in Figure 2.

Given a single data point and its velocity, the probability density function of a DND is defined as

$$p\left([\boldsymbol{x},\boldsymbol{v}]\,|t,\boldsymbol{\mu},\boldsymbol{\Sigma}\right) = k e^{-\frac{1}{2}(\boldsymbol{x}+t\boldsymbol{v}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}+t\boldsymbol{v}-\boldsymbol{\mu})}, \quad (1)$$
$$k = \frac{1}{\sqrt{(2\pi)^d\,|\boldsymbol{\Sigma}|}}.$$

The scalar parameter t determines how far a point is projected along its velocity vector. The parameter t has to be zero or positive. We assume that t is distributed according to a normal distribution. The constraint $t \geq 0$ leads to a truncated normal distribution, a normal distribution whose lower tail is truncated at zero

$$p\left(t\middle|\mu_t,\sigma_t^2\right) = \frac{1}{\sqrt{(2\pi)}\sigma_t}\frac{1}{1-\Phi(-\frac{\mu_t}{\sigma_t})}e^{-\frac{(t-\mu_t)^2}{2\sigma_t^2}}. \quad (2)$$

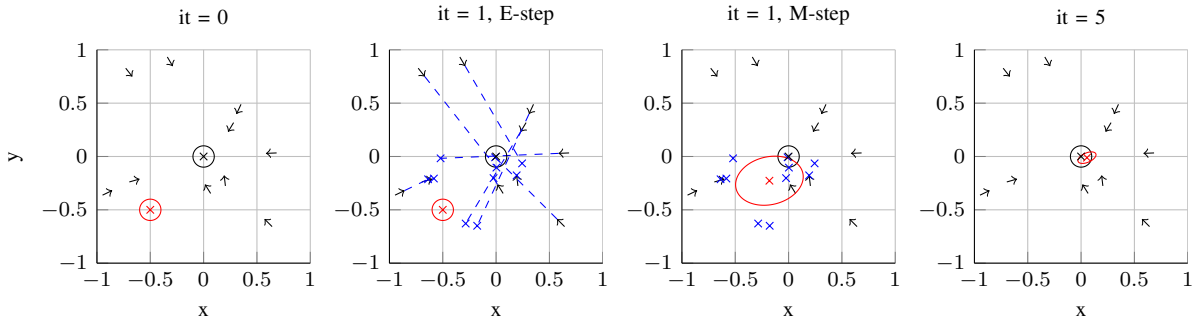


Fig. 2: Illustration of the DND and the learning algorithm. The black arrows indicate the data points and the velocity vectors. The data points are drawn uniformly from the shown grid. The direction/velocity is determined by drawing points from the black normal distribution and scaling the difference vector to the current position to a fixed value. The EM algorithm for learning the DND parameters estimates a target on the grid that fits best to all data points and the corresponding velocity vectors. The red ellipsoid shows the initial and current guesses for the target. In the E-step of the algorithm, for each data point a projection along the velocity vector is computed that fits best to the current estimate of the target (blue dashed lines). In the M-step, a new guess for the distribution parameter is estimated.

Here, Φ is the cumulative distribution function of a normal distribution with standard deviation one and mean zero. By rearranging (1), the equation can also be interpreted as posterior distribution of t

$$p\left(t\middle|\left[\boldsymbol{x},\boldsymbol{v}\right],\boldsymbol{\mu},\boldsymbol{\Sigma}\right) = \frac{1}{\sqrt{(2\pi)}\sigma_t}\frac{1}{1-\Phi(-\frac{\mu_t}{\sigma_t})}e^{-\frac{(t-\mu_t)^2}{2\sigma_t^2}}, \quad (3)$$
$$\mu_t = \frac{\boldsymbol{v}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}-\boldsymbol{x})}{\boldsymbol{v}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{v}},$$
$$\sigma_t = (\boldsymbol{v}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{v})^{-\frac{1}{2}}.$$

The joint probability is defined as

$$p\left(\left[\boldsymbol{x},\boldsymbol{v}\right],t\middle|\boldsymbol{\mu},\boldsymbol{\Sigma},\mu_t,\sigma_t^2\right) = p\left(\left[\boldsymbol{x},\boldsymbol{v}\right]\middle|t,\boldsymbol{\mu},\boldsymbol{\Sigma}\right) p\left(t\middle|\mu_t,\sigma_t^2\right). \quad (4)$$

We omitted to explicitly state that (2) and (4) are zero for $t < 0$. For learning the parameters of the distribution for a given set of N points $\boldsymbol{X} \in \mathbb{R}^{d\times N}$ and the corresponding velocities $\boldsymbol{V} \in \mathbb{R}^{d\times N}$, we use a maximum likelihood approach. The distributions of the scale parameters t_i for data points i are not a part of the data and therefore unknown. The log-likelihood function for the data set is defined as

$$\log p\left(\left[\boldsymbol{X},\boldsymbol{V}\right]\middle|\boldsymbol{\mu},\boldsymbol{\Sigma}\right) = \sum_{i=1}^{N}\int_0^\infty \log p\left(\left[\boldsymbol{x}_i,\boldsymbol{v}_i\right],t_i\middle|\boldsymbol{\mu},\boldsymbol{\Sigma},\mu_t^{(i)},\sigma_t^{2(i)}\right) dt_i.$$

Here, we assumed that the values t_i are independent of each other. The integration over t_i is intractable as the variables depend on the unknown parameters $\mu_t^{(i)}$ and $\sigma_t^{(i)}$. Therefore, we treat them as latent variables and derive an EM algorithm for inferring the parameters of our distribution.

A. Expectation-Maximization

In the Expectation step of the algorithm, we use the current parameter values $\boldsymbol{\theta}^{\mathrm{old}} = \{\boldsymbol{\mu}^{\mathrm{old}}, \boldsymbol{\Sigma}^{\mathrm{old}}\}$ to estimate the posterior distribution of the latent variables $p(\boldsymbol{t}|\boldsymbol{X},\boldsymbol{V},\boldsymbol{\theta}^{\mathrm{old}})$. As the t_i's are independent of each other, the estimation can be done for each data point separately as in (3). In the Maximization step, we estimate the new parameters $\boldsymbol{\theta}^{\mathrm{new}}$ by maximizing the expectation of the complete-data loglikelihood under the posterior of the latent variables

$$\begin{aligned}\boldsymbol{\theta}^{\mathrm{new}} &= \arg\max_{\boldsymbol{\Theta}} \mathbb{E}_{\boldsymbol{t}}\left[\log p\left(\left[\boldsymbol{X},\boldsymbol{V}\right],\boldsymbol{t}\middle|\boldsymbol{\theta},\boldsymbol{\mu}_t,\boldsymbol{\sigma}_t\right)\right], \\ &= \arg\max_{\boldsymbol{\Theta}} \sum_{i=1}^{N}\int_0^\infty p\left(t_i\middle|\left[\boldsymbol{x}_i,\boldsymbol{v}_i\right],\boldsymbol{\theta}\right) \\ &\qquad \log p\left(\left[\boldsymbol{x}_i,\boldsymbol{v}_i\right],t_i\middle|\boldsymbol{\theta},\mu_t^{(i)},\sigma_t^{2(i)}\right) dt_i. \quad (5)\end{aligned}$$

As the integral is evaluated for each data point separately, we omit the indices for the solution of the integral in the following. The integral evaluates to

$$\begin{aligned}&\int_0^\infty p\left(t\middle|\left[\boldsymbol{x},\boldsymbol{v}\right],\boldsymbol{\theta}\right)\log p\left(\left[\boldsymbol{x},\boldsymbol{v}\right],t\middle|\boldsymbol{\theta},\mu_t,\sigma_t\right) dt \\ &= \int_0^\infty c_1 e^{-\frac{(t-\mu_t)^2}{2\sigma_t^2}}\left(c_2(\boldsymbol{\Theta})t^2 + c_3(\boldsymbol{\Theta})t + c_4(\boldsymbol{\Theta})\right) dt \\ &= c_1\left(c_2(\boldsymbol{\Theta})d_2 + c_3(\boldsymbol{\Theta})d_3 + c_4(\boldsymbol{\Theta})d_4\right). \quad (6)\end{aligned}$$

The values of the constants can be found in the Appendix. Note that c_2, c_3, and c_4 are constants only for the integration, as they depend on the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ which we want to estimate. Now that the integral has been evaluated, the result can be used to find the parameters according to (5). As the constants c_2 to c_4 depend on our parameters, we compute the derivatives of (6) with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}^{-1}$ and set them to zero. Now we work again on the entire data set, so we will use the indices again.

$$\begin{aligned}&\frac{\partial\mathbb{E}_{\boldsymbol{t}}\left[\log p\left(\left[\boldsymbol{X},\boldsymbol{V}\right],\boldsymbol{t}\middle|\boldsymbol{\theta},\boldsymbol{\mu}_t,\boldsymbol{\sigma}_t\right)\right]}{\partial\boldsymbol{\mu}} \\ &= \sum_{i=1}^{N} c_1^{(i)}\left(d_2^{(i)}\frac{\partial c_2^{(i)}}{\partial\boldsymbol{\mu}} + d_3^{(i)}\frac{\partial c_3^{(i)}}{\partial\boldsymbol{\mu}} + d_4^{(i)}\frac{\partial c_4^{(i)}}{\partial\boldsymbol{\mu}}\right) \\ &= \boldsymbol{\Sigma}^{-1}\sum_{i=1}^{N} c_1^{(i)}\left(d_3^{(i)}\boldsymbol{v}_i - d_4^{(i)}\boldsymbol{\mu} + d_4^{(i)}\boldsymbol{x}_i\right) = \boldsymbol{0}.\end{aligned}$$

Multiplying by $\boldsymbol{\Sigma}$ from the left and rearranging leads to the solution

$$\boldsymbol{\mu} = \frac{\sum_{i=1}^{N} c_1^{(i)}d_4^{(i)}\boldsymbol{x}_i + c_1^{(i)}d_3^{(i)}\boldsymbol{v}_i}{\sum_{i=1}^{N} c_1^{(i)}d_4^{(i)}}. \quad (7)$$

Note that the cumulative distribution function Φ is closely related to the error function erf by the relation

$$\Phi\left(-\frac{\mu_t}{\sigma_t}\right) = \frac{1}{2}\left(1 - \operatorname{erf}\left(\frac{\mu_t}{\sqrt{2}\sigma_t}\right)\right).$$

Therefore, all constants $c_1 d_j$ could be further simplified. For example, $c_1 d_4$ equals one (proof skipped here) and so (7) can also be written as

$$\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N}\left(\boldsymbol{x}_i + c_1^{(i)} d_3^{(i)} \boldsymbol{v}_i\right).$$

For estimating the covariance matrix $\boldsymbol{\Sigma}$, we compute the derivative with respect to its inverse $\boldsymbol{\Sigma}^{-1}$.

$$\begin{aligned}
&\frac{\partial \mathbb{E}_{\boldsymbol{t}}\left[\log p\left([\boldsymbol{X}, \boldsymbol{V}], \boldsymbol{t} \middle| \boldsymbol{\theta}, \boldsymbol{\mu}_t, \boldsymbol{\sigma}_t\right)\right]}{\partial \boldsymbol{\Sigma}^{-1}} \\
&= \sum_{i=1}^{N} c_1^{(i)} \left(d_2^{(i)} \frac{\partial c_2^{(i)}}{\partial \boldsymbol{\Sigma}^{-1}} + d_3^{(i)} \frac{\partial c_3^{(i)}}{\partial \boldsymbol{\Sigma}^{-1}} + d_4^{(i)} \frac{\partial c_4^{(i)}}{\partial \boldsymbol{\Sigma}^{-1}} \right) \\
&= \frac{1}{2}\sum_{i=1}^{N} c_1^{(i)} \Big(- d_2^{(i)} \boldsymbol{v}_i \boldsymbol{v}_i^{\mathrm{T}} + d_3^{(i)} \boldsymbol{v}_i (\boldsymbol{\mu} - \boldsymbol{x}_i)^{\mathrm{T}} \\
&\quad + d_3^{(i)} (\boldsymbol{\mu} - \boldsymbol{x}_i) \boldsymbol{v}_i^{\mathrm{T}} - d_4^{(i)} (\boldsymbol{\mu} - \boldsymbol{x}_i)(\boldsymbol{\mu} - \boldsymbol{x}_i)^{\mathrm{T}} \\
&\quad + d_4^{(i)} \boldsymbol{\Sigma} \Big) = \boldsymbol{0}.
\end{aligned}$$

Rearranging then leads to the solution

$$\begin{aligned}
\boldsymbol{\Sigma} = \frac{1}{N}\sum_{i=1}^{N} c_1^{(i)} \Big(& d_2^{(i)} \boldsymbol{v}_i \boldsymbol{v}_i^{\mathrm{T}} - d_3^{(i)} \boldsymbol{v}_i (\boldsymbol{\mu} - \boldsymbol{x}_i)^{\mathrm{T}} \\
& - d_3^{(i)} (\boldsymbol{\mu} - \boldsymbol{x}_i) \boldsymbol{v}_i^{\mathrm{T}} + d_4^{(i)} (\boldsymbol{\mu} - \boldsymbol{x}_i)(\boldsymbol{\mu} - \boldsymbol{x}_i)^{\mathrm{T}} \Big).
\end{aligned}$$

B. Extension for Orientations

Equation (1) contains a distance vector between the current data point projected along the velocity vector $\boldsymbol{x} + t\boldsymbol{v}$ and the mean of the target $\boldsymbol{\mu}$. As such a distance vector is not a suitable distance measure for 3D orientations, we have to explicitly derive a DND for orientations. Let us first neglect the uncertainty and assume that we know the target orientation. Given a current data point (a start orientation) and an angular velocity, we want to rotate the start orientation following the angular velocity so that the distance to the target orientation becomes minimal. The rotation matrix $\boldsymbol{R}_{\mathrm{FI}}$ describes the target orientation in the inertia frame. The rotation matrix $\boldsymbol{R}_{\mathrm{VI}}(t) = \boldsymbol{R}_{\mathrm{VS}}(t)\boldsymbol{R}_{\mathrm{SI}}$ describes the start orientation rotated with the angle t around the angular velocity axis, also given in the inertia frame. The distance between both orientations is defined as the axis angle $\theta(t)$ of the relative transformation $\boldsymbol{R}_{\mathrm{FV}} = \boldsymbol{R}_{\mathrm{FI}}(\boldsymbol{R}_{\mathrm{VI}})^{\mathrm{T}}$

$$\begin{aligned}
\theta(t) &= \cos^{-1}\left(\frac{1}{2}\left(\operatorname{Tr}(\boldsymbol{R}_{\mathrm{FV}}) - 1\right)\right) \\
&= \cos^{-1}\left(\frac{1}{2}\left(a\cos(t) + b\sin(t) + c - 1\right)\right). \quad (8)
\end{aligned}$$

The constants a, b and c can be derived straightforwardly and are omitted here due to space constraints. To find the minimum angle, we compute the derivative of (8) with respect to t and set it to zero

$$\begin{aligned}
\frac{\partial \theta}{\partial t} &= \frac{0.5a\sin(t) - 0.5b\cos(t)}{\sqrt{1 - \left(\frac{1}{2}\left(a\cos(t) + b\sin(t) + c - 1\right)\right)^2}} \\
&= 0 \quad \Rightarrow \quad t = \operatorname{atan2}(b, a). \quad (9)
\end{aligned}$$

As a next step, we want to define a difference vector that can be written in the form $\boldsymbol{x} + t\boldsymbol{v} - \boldsymbol{\mu}$, so that we can use the same EM algorithm as in the previous section for estimating the target orientation. Due to the trigonometric dependency on t, some approximations have to be made. First, we write the matrices as vectors $\boldsymbol{r} \in \mathbb{R}^{9\times 1}$. Then, we define a difference vector

$$\begin{aligned}
\boldsymbol{r}_{\mathrm{VI}}(t) - \boldsymbol{r}_{\mathrm{FI}} &\approx \boldsymbol{x} + t\boldsymbol{v} - \boldsymbol{\mu}, \\
\boldsymbol{r}_{\mathrm{VI}}(t) &= \boldsymbol{d}\cos(t) + \boldsymbol{e}\sin(t) + \boldsymbol{f} \\
&\approx \boldsymbol{d}(\cos(\alpha) - \sin(\alpha)(t - \alpha)) \\
&\quad + \boldsymbol{e}(\sin(\alpha) + \cos(\alpha)(t - \alpha)) + \boldsymbol{f}, \\
\boldsymbol{x} &= \alpha\cos(\alpha)(-\boldsymbol{d} - \boldsymbol{e}) \\
&\quad + \alpha\sin(\alpha)(-\boldsymbol{d} + \boldsymbol{e}) + \boldsymbol{f}, \\
\boldsymbol{v} &= \cos(\alpha)(d + e) + \sin(\alpha)(-d + e), \\
\boldsymbol{\mu} &= \boldsymbol{r}_{\mathrm{FI}}.
\end{aligned}$$

Again, we do not explicitly derive the constants $\boldsymbol{d}$, $\boldsymbol{e}$ and $\boldsymbol{f}$. We approximated the sine and cosine terms with a first-order Taylor approximation around α. Now our difference vector can be written in the form $\boldsymbol{x} + t\boldsymbol{v} - \boldsymbol{\mu}$ and we can estimate the target orientation with the same algorithm as in the previous section. Still, some details have to be taken care of. Firstly, the approximation can be arbitrarily bad depending on how much α deviates from t. Therefore, we make use of a trick. If we neglect the uncertainty $\boldsymbol{\Sigma}$, we can take the current estimate of the mean $\boldsymbol{\mu}$ to compute $\tilde{t}$ according to (9). Then, $\tilde{t}$ can be used to recompute the approximation constants at every iteration by developing the Taylor series around it instead of making the approximation only once. By doing that, we know that we develop the Taylor series about a value close to the true t, leading to a good approximation. Secondly, we ignored the constraint that the resulting vector (reshaped to a matrix) has to describe a proper rotation. Therefore, we have to find the closest true rotation matrix to our matrix, which can be done for example as in [20].

C. DNDs as HMM Output Emissions

Two things have to be considered when using DNDs as HMM output emissions for the task decomposition. Firstly, there is no noise considered in the model. As the trajectories demonstrated by the teacher are usually no straight lines, this may lead to an inaccurate estimation of the DND parameters. We therefore suggest to weight data points according to their position in a segment for learning. Points closer to the end of a possible segment are assigned higher weights, so that they have a larger influence on the resulting mean of the distribution. Secondly, if the robot is not moving and

(a) (b) (c)

Fig. 3: The three different experimental setups for the light bulb unscrewing task. For each setup, the light bulb holder and box were put to different locations on the two tables.

MP	Description	Frame	Control Variables
●	Approach Light Bulb	Light Bulb	Position xyz
●	Pull Bulb	Light Bulb	Position xy Force z
●	Lift Bulb	Light Bulb	Position xyz
●	Approach Box	Box	Position xyz
●	Approach Final Position	Box	Position xyz
●	Screw	World	Axis Angles
●	Unscrew	World	Axis Angles
●	Open Fingers	World	Finger Joints
●	Close Fingers	World	Finger Joints

TABLE I: Resulting coordinate frame and control variables for each MP. The first three MPs control the end-effector relative to the light bulb position, while the latter two at the end of the demonstration control the robot relative to the box. Only the MP that is active during unscrewing applies a force, as the teacher was pulling the light bulb during this phase of the task. Two MPs control the orientation of the end-effector and the fingers, respectively.

the velocity is close to zero, the algorithm may run into numerical issues. Therefore, we set t to zero if the norm of the velocity is below a threshold.

IV. Experiments

For evaluating our approach, we performed kinesthetic demonstrations on a gravity compensated seven degrees of freedom (DOF) Barrett WAM robot with a four DOF hand. In previous work, we demonstrated the task of unscrewing a light bulb on the same robot [21]. We segmented the demonstrations manually and chose the task decomposition beforehand. Then, we showed that our system successfully learned how to sequence the MPs in order to reproduce the demonstrated task. In this paper, we use data from these previous experiments and show that our approach finds a task decomposition that is similar to the one we defined manually.

A. Experimental Setup

For the unscrewing task, the human teacher demonstrated the following sequence of subtasks. First, he approached the light bulb with the end-effector. Then, the robot's hand was closed by pressing a key on a keyboard. For simplicity, the grasp was predefined, even though teaching grasps is possible with our approach. Next, the teacher rotates the wrist of the robot arm and unscrews the bulb. During this movement, he also pulls the light bulb (by applying a force along the z-axis), to test whether it is still in its holder. After turning the light bulb, the fingers are opened (again, this movement is started by pressing a key) and the wrist is rotated back. This unscrewing cycle is repeated until the light bulb gets loose. Then, the light bulb is pulled out of the holder and the end-effector is moved to a box where the hand is opened again.

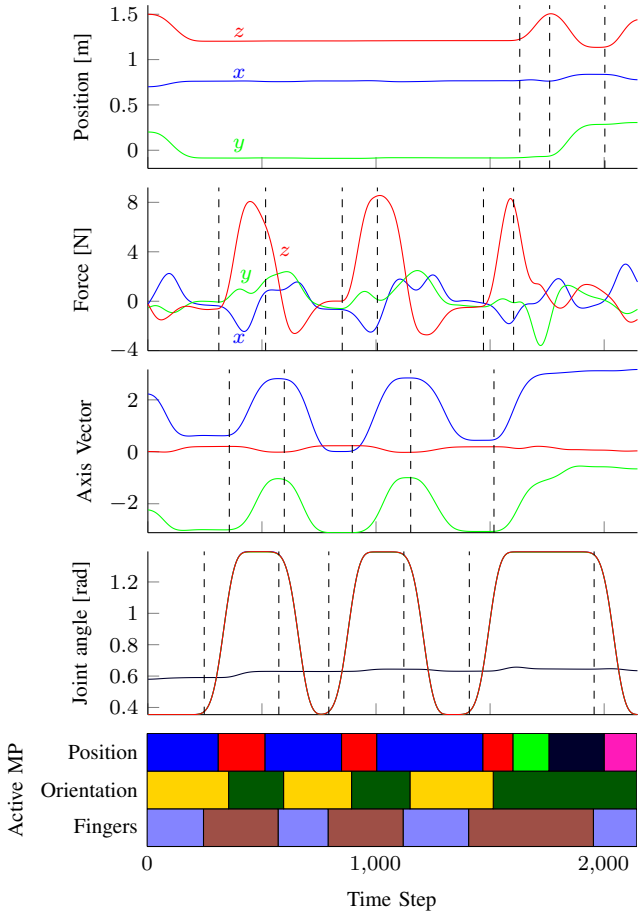
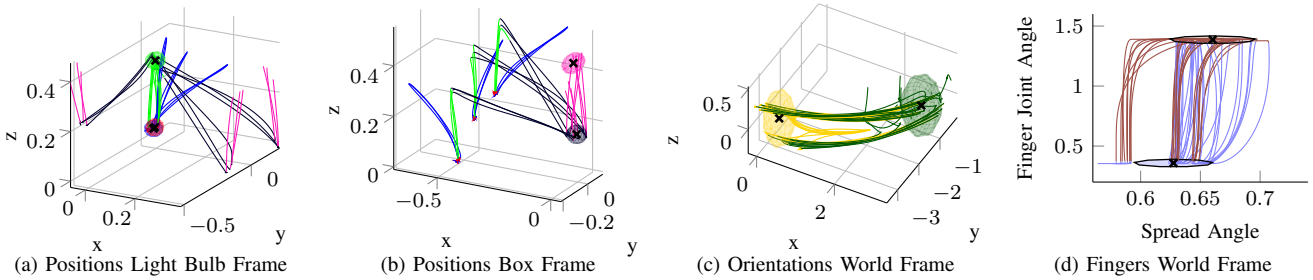


Fig. 4: Task-decomposition over time for one of the nine demonstrations. From the top, the plots show the position of the end-effector, the measured forces at the wrist, the orientation of the end-effector and the finger joint angles, respectively. The dashed lines indicate the zero-velocity crossings and contact changes. The plot on the bottom shows the most likely MP for each point in time. All MPs can be associated with a meaningful description (see Table I).

For the demonstrations, we put the light bulb holder and the box to three different positions and performed three kinesthetic demonstrations for each setup. The setups are depicted in Figure 3. We defined three coordinate frames. The world frame, the light bulb frame and the box frame and thus, the decomposition was performed with three task-spaces ($N_{\mathrm{TS}} = 3$). The kinesthetic demonstration data was recorded with 200Hz. As the force-torque measurements are quite noisy, we filtered the data (also the joint angle data) using a Hanning filter with a window size of 100.

As position/force of the end-effector, its orientation and the fingers can be controlled independently of each other and our MP framework allows the activation of multiple MPs at the same time (see [1]), we also performed the decomposition for each of them independently. The overall task-decomposition therefore results in three active MPs for each time step. For each decomposition, only the relevant ZVCs where used as potential cuts between the segments (e.g., only the ZVCs of the finger joint angles were used for the decomposition of the MPs controlling the fingers).



(a) Positions Light Bulb Frame (b) Positions Box Frame (c) Orientations World Frame (d) Fingers World Frame

Fig. 5: Task-space trajectories for all nine demonstrations. Plots (a) and (b) show the end-effector positions in the light bulb coordinate frame and box coordinate frame, respectively. The colors indicate to which MP each segment is assigned to. The markers correspond to the mean $\boldsymbol{\mu}$ of the MP targets and the ellipsoids indicate their covariance matrices $\boldsymbol{\Sigma}$. Note that the position target of the blue MP is hard to see because it coincides with the target of the red MP. The targets are only plotted in the coordinate frame that was assigned to each MP. Plots (c) and (d) show the orientations of the end-effector and the finger configurations, respectively (both in the world frame). As for this task all three fingers were aligned equally, only the joint angle of one finger is shown in (d).

MP	x	y	z
●	0.08	-1.43	-0.51
●	-1.08	0.30	**7.12**
●	1.08	-1.35	0.02
●	0.06	0.01	-0.17
●	-0.24	-0.85	-0.88

TABLE II: Resulting target forces in Newton in the corresponding frames of the position MPs. Only the red MP is force-controlled along the z-axis because the desired force is large enough.

B. Results

Table I shows the MPs resulting from the decomposition of the kinesthetic demonstrations. Our method found four MPs controlling the position of the end-effector and one MP that is controlling the position of the end-effector along the x- and y-axes, while applying a force along the z-axis. Additionally, two MPs control the orientation of the end-effector and two MPs the finger configuration. Figure 4 depicts the most likely MP for each point in time for one of the nine demonstrations. Except for small time gaps between changes of position, orientation and/or finger MPs, the MP sequences exactly reflect the subtask sequence the teacher performed and thus we can associate each MP with a meaningful description in Table I. Note that the time gaps are not incorrect but instead reflect the behavior of the teacher. For instance, it is not possible to unscrew the bulb before closing the fingers. Therefore, the teacher usually waits for a short moment until he or she is sure the fingers are closed before starting to unscrew, causing the aforementioned time gaps in the data.

Figure 5 shows the task-space trajectories of all nine demonstrations. The plots indicate that even though the number of unscrewing repetitions varied over the demonstrations, all demonstrations were decomposed in a similar way. As all subtasks performed by the teacher can be described relative to either the light bulb or the box, the world frame was not chosen for any MP controlling the position of the end-effector. During the demonstrations, the teacher was standing at a fixed position and aligned the orientations equally for all demonstrations. However, the objects and thus also their corresponding coordinate frames were rotated for the different setups. As a result, our algorithm chose the world frame for all orientation and finger configuration MPs.

Table II shows the resulting target forces from the DNDs for all position MPs. Only for the red MP the target force along the z-axis is large enough so that the MP is force-controlled along this axis. The MP corresponds to the teacher pulling the light bulb while unscrewing. When reproducing the task, this force leads to an acceleration of the robot's arm as soon as the light bulb gets loose, enabling the system to detect the loose bulb without any external sensors. The drawback is that we cannot detect whether the light bulb is still in the hand of the robot or if it slipped during reproduction.

After decomposing the demonstrations, we evaluated the resulting MPs by activating them manually on the real robot. In previous work [21], we learned the activations for manually labelled demonstrations and a predefined task-decomposition. We consider it future work to combine both approaches. By activating the MPs in the correct order, we were successfully able to reproduce the task on the robot.

C. Discussion

The results presented in the previous section show that our method segments the demonstrations properly and finds a meaningful task-decomposition. In fact, the resulting MP sequences exactly reflect the necessary sequence of subtasks we expected would be necessary to perform the task (see task description in Section IV-A). BIC proved to be a reliable criterion for choosing an appropriate number of MPs. Still, we have to evaluate our method also on different tasks to ensure that the criterion yields good results in general.

In the previous section, we did not mention that we scaled the force data by a factor before training. The scale factor reflects an important issue occurring in kinesthetic demonstrations. While it is easy to guide a gravity compensated robot to a desired position, applying desired forces is difficult. In the demonstrations, we tried to apply always the same force when pulling the light bulb. Still, the values were roughly in a range of 5 to $15\,\mathrm{N}$. As a consequence, we state that from the force data of a kinesthetic demonstration, it is only clearly distinguishable whether the teacher wanted to push or pull in a certain direction or did not apply a force at all. We therefore scaled the forces to reduce their influence on the decomposition. In the experiments, the factor was

set to $1/40$, so that $4\,\mathrm{N}$ difference correspond to a position difference of $10\,\mathrm{cm}$. We evaluated different scale factors and observed that the segmentation result is robust for a broad range of values ($1/20$ to $1/80$ yield the same results). Hence, our method requires, at least for this task, no accurate fine-tuning of parameters. In general, the scale factor influences the importance of the force. If the factor is too small, the force is ignored completely (and, for example, the blue and red MPs would be merged to one MP). If it is too large, the force overpowers the position (e.g., it may happen that some segments describing the unscrewing movement would be assigned to different MPs). To avoid having a fixed threshold, we consider it future work to estimate the importance of the force from the demonstration data. A possibility is to learn parameters weighting the importance of the force within the hybrid force-position controller.

V. Conclusion and Future Work

We presented a novel concept for decomposing kinesthetic demonstrations of sequential force interaction tasks into a set of MPs. To capture the sequential properties of the task, we model the demonstrations with a HMM, where states correspond to the individual MPs. The main contribution of this paper is a novel probability distribution (DND). Mixtures of this distribution are used as output emissions of the HMM, which allows to simultaneously determine the most likely sequence of MPs as well as their composition, i.e., their coordinate frames, control variables and target coordinates from the demonstration data. To the best of our knowledge this is the first concept to do this simultaneously. The resulting sequences resemble very closely the natural structure of the task. While this paper's focus was on the decomposition of the kinesthetic demonstrations, future work will target on combining this concept with previous work on learning the MP transition behavior and reproducing the learned skills [21], [22].

Appendix

$$c_1 = \frac{1}{\sqrt{(2\pi)}\sigma_t}\frac{1}{1-\Phi(-\frac{\mu_t}{\sigma_t})}, \quad c_2 = -\frac{1}{2}\boldsymbol{v}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{v} - \frac{1}{2\sigma_t^2},$$

$$c_3 = \frac{1}{2}\boldsymbol{v}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}-\boldsymbol{x}) + \frac{1}{2}(\boldsymbol{\mu}-\boldsymbol{x})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{v} + \frac{\mu_t}{\sigma_t^2},$$

$$c_4 = -\frac{1}{2}\log\left((2\pi)^{d+1}\sigma_t^2\,|\boldsymbol{\Sigma}|\right) - \log\left(1-\Phi\left(-\frac{\mu_t}{\sigma_t}\right)\right) - \frac{1}{2}(\boldsymbol{\mu}-\boldsymbol{x})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}-\boldsymbol{x}) - \frac{\mu_t^2}{2\sigma_t^2}$$

$$d_2 = \mu_t\sigma_t^2 e^{-\frac{\mu_t^2}{2\sigma_t^2}} + \sqrt{\frac{\pi}{2}}\sigma_t\left(\mu_t^2+\sigma_t^2\right)\left(1+\mathrm{erf}\left(\frac{\mu_t}{\sqrt{2}\sigma_t}\right)\right),$$

$$d_3 = \sigma_t^2 e^{-\frac{\mu_t^2}{2\sigma_t^2}} + \sqrt{\frac{\pi}{2}}\mu_t\sigma_t\left(1+\mathrm{erf}\left(\frac{\mu_t}{\sqrt{2}\sigma_t}\right)\right),$$

$$d_4 = \sqrt{\frac{\pi}{2}}\sigma_t\left(1+\mathrm{erf}\left(\frac{\mu_t}{\sqrt{2}\sigma_t}\right)\right).$$

References

[1] T. Luksch, M. Gienger, M. Muehlig, and T. Yoshiike, "Adaptive movement sequences and predictive decisions based on hierarchical dynamical systems," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2012.

[2] S. Niekum, S. Osentoski, G. D. Konidaris, S. Chitta, B. Marthi, and A. G. Barto, "Learning grounded finite-state representations from unstructured demonstrations," *International Journal of Robotics Research*, vol. 34, no. 2, pp. 131–157, 2015.

[3] R. Lioutikov, G. Neumann, G. Maeda, and J. Peters, "Probabilistic segmentation applied to an assembly task," in *Int. Conf. Humanoid Robots*, 2015.

[4] P. Ranchod, B. Rosman, and G. Konidaris, "Nonparametric bayesian reward segmentation for skill discovery using inverse reinforcement learning," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2015.

[5] B. Michini, *Bayesian Nonparametric Reward Learning from Demonstration*. PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, Cambridge MA, August 2013.

[6] S. Krishnan, A. Garg, S. Patil, C. Lea, G. Hager, P. Abbeel, and K. Goldberg, "Transition state clustering: Unsupervised surgical trajectory segmentation for robot learning," in *International Symposium on Robotics Research*, 2015.

[7] D. Endres, A. Christensen, L. Omlor, and M. A. Giese, "Emulating human observers with bayesian binning: Segmentation of action streams," *ACM Trans. Applied Perception*, vol. 8, no. 3, pp. 16:1–16:12, 2011.

[8] K. Gräve and S. Behnke, "Incremental action recognition and generalizing motion generation based on goal-directed features," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2012.

[9] G. Konidaris, S. Kuindersma, R. Grupen, and A. Barto, "Robot learning from demonstration by constructing skill trees," *Int. Journal of Robotics Research*, vol. 31, no. 3, pp. 360–375, 2012.

[10] D. Kulić, C. Ott, D. Lee, J. Ishikawa, and Y. Nakamura, "Incremental learning of full body motion primitives and their sequencing through human motion observation," *Int. Journal of Robotics Research*, vol. 31, no. 3, pp. 330–345, 2012.

[11] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Int. Conf. Machine Learning*, 2001.

[12] A. Baisero, Y. Mollard, M. Lopes, M. Toussaint, and I. Lutkebohle, "Temporal segmentation of pair-wise interaction phases in sequential manipulation demonstrations," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2015.

[13] L. Rozo, D. Bruno, S. Calinon, and D. G. Caldwell, "Learning optimal controllers in human-robot cooperative transportation tasks with position and force constraints," in *Int. Conf. Intelligent Robots and Systems*, 2015.

[14] F. J. Abu-Dakka, B. Nemec, J. A. Jørgensen, T. R. Savarimuthu, N. Krüger, and A. Ude, "Adaptation of manipulation skills in physical contact with the environment to reference force profiles," *Autonomous Robots*, vol. 39, no. 2, pp. 199–217, 2015.

[15] F. Steinmetz, A. Montebelli, and V. Kyrki, "Simultaneous kinesthetic teaching of positional and force requirements for sequential in-contact tasks," in *IEEE-RAS Int. Conf. Humanoid Robots*, 2015.

[16] A. Ureche, K. Umezawa, Y. Nakamura, and A. Billard, "Task parameterization using continuous constraints extracted from human demonstrations," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1458–1471, 2015.

[17] J. Kober, M. Gienger, and J. Steil, "Learning movement primitives for force interaction tasks," in *IEEE Int. Conf. Robotics and Automation*, 2015.

[18] P. Pastor, M. Kalakrishnan, L. Righetti, and S. Schaal, "Towards associative skill memories," in *IEEE-RAS Int. Conf. Humanoid Robots*, 2012.

[19] J. R. Flanagan, M. C. Bowman, and R. S. Johansson, "Control strategies in object manipulation tasks," *Current Opinion in Neurobiology*, vol. 16, no. 6, pp. 650–659, 2006.

[20] N. Higham, "Matrix nearness problems and applications," in *Applications of Matrix Theory*, pp. 1–27, Oxford University Press, 1989.

[21] S. Manschitz, J. Kober, M. Gienger, and J. Peters, "Learning movement primitive attractor goals and sequential skills from kinesthetic demonstrations," *Robotics and Autonomous Systems*, vol. 74, pp. 97–107, 2015.

[22] S. Manschitz, J. Kober, M. Gienger, and J. Peters, "Probabilistic progress prediction and sequencing of concurrent movement primitives," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2015.