Movement Segmentation and Recognition for Imitation Learning

Franziska Meier[1] fmeier@usc.edu
Evangelos Theodorou[2] etheodor@cs.washington.edu
Stefan Schaal[1,3] sschaal@usc.edu

[1]Computational Learning and Motor Control Lab, University of Southern California, Los Angeles
[2] Department of Computer Science and Engineering, University of Washington, Seattle
[3]Max-Planck-Institute for Intelligent Systems, Tübingen, Germany

Abstract

In human movement learning, it is most common to teach constituent elements of complex movements in isolation, before chaining them into complex movements. Segmentation and recognition of observed movement could thus proceed out of this existing knowledge, which is directly compatible with movement generation. In this paper, we address exactly this scenario. We assume that a library of movement primitives has already been taught, and we wish to identify elements of the library in a complex motor act, where the individual elements have been smoothed together, and, occasionally, there might be a movement segment that is not in our library yet. We employ a flexible machine learning representation of movement primitives based on learnable nonlinear attractor system. For the purpose of movement segmentation and recognition, it is possible to reformulate this representation as a controlled linear dynamical system. An Expectation-Maximization algorithm can be developed to estimate the open parameters of a movement primitive from the library, using as input an observed trajectory piece. If no matching primitive from the library can be found, a new primitive is created. This process allows a straightforward sequential segmentation of observed movement into known and new primitives, which are suitable for robot imitation learning. We illustrate our approach with synthetic examples and data collected from human movement.

Appearing in Proceedings of the 15^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2012, La Palma, Canary Islands. Volume XX of JMLR: W&CP XX.

1 Introduction

When trying to automatically observe and interpret complex human movement sequences, the ability to segment complex movements into simple action units plays an important role. While in many vision applications like surveillance the focus lies solely on identifying and classifying these action units, in imitation learning we also want to be able to reproduce the observed movement. Thus, when performing movement segmentation and recognition for the purpose of imitation learning it is important to use a movement representation that is suitable for both recognition and reproduction of movement.

Our work addresses exactly this issue. The larger goal of our research is to bootstrap autonomous learning robots by imitation learning, i.e., to teach robots by demonstrating motor skills, which the robot can subsequently refine by autonomous trial-and-error learning. For this, we adopt the assumption that complex movement skills are composed from action units, also called movement primitives [1]. Furthermore, we assume that a library of movement primitives exists, that comprises useful motion segments that have been previously taught in isolation. This is in contrast to many existing approaches, e.g., based on statistical methods [2, 3]. Typically it is hard to extract behaviorally meaningful movement primitives from observed movement by just using statistical analysis. As a result, it is often not straight forward to generate movement given a desired task with such primitives.

Thus, in this research, our goal is to recognize primitives from a library in a complex movement sequence. A typical problem in this scenario is that temporally adjacent primitives have been smoothed together, such that clear demarcation points for segmentation are not easy to find. Another issue arises when encountering unknown primitives, that have not been stored in the library. This work presents a segmentation framework that can handle both of these issues.

After giving a brief literature review in the next section, present our movement segmentation strategy 2. In Section 3 and 4 we introduce the mathematical and algorithmic details of our approach. Finally, Section 5 presents an evaluation of our approach.

1.1 Related Work

As indicated in the introduction, movement segmentation algorithms have to deal with different constraints and requirements depending on the application. In general the approaches to movement generation can be grouped into two main categories. The first category consists of methods that perform segmentation without the use of pre-trained motion primitive models, [4, 5, 6, 7]. As discussed above, the lack of a library to perform the segmentation usually leads to extraction of primitives that are not easily usable to generate movements for a desired task. Furthermore, these approaches utilize representations of movement primitives that usually do not facilitate movement reproduction on a robot.

The second major category comprises of approaches that use pre-trained motion models and perform movement segmentation with simultaneous movement recognition, an approach prevalent in the vision community, [8, 9, 10]. More details on movement segmentation in the vision community is given in [11]. While these approaches are using some form of library to segment and recognize movements, they have been proposed for vision applications and thus do not consider the capability of movement reproduction as a requirement. Thus, the representations for activity understanding only subserve the subsequent classification algorithms, and do not relate to movement generation on a robotic system.

Our approach as well as the work described in [12] falls into the second category, as we use a library of movement primitives to perform the segmentation and recognition of movement sequences. In [12] one could augment the proposed generative model with pre trained movement primitives in the form of dynamical systems. However, it is not clear whether an instance of a primitive with different orientation and scaling could be recognized or generated by the underlying representation. We differ by utilizing a movement primitive representation that can both recognize and generate a family of movement trajectories for a given primitive. Furthermore, we show that this representation helps identify good segmentation point choices in case we encounter an unknown primitive that cannot be matched to a primitive in the library. As a result we are not only able to segment movement sequences with known movement primitives, but are also capable of updating the library with new primitives.

2 Segmentation using a Movement Primitive Library

This work approaches the problem of movement segmentation by assuming that a library of movement primitives $\mathcal{L} = \{\Theta^{(1)}, \ldots, \Theta^{(m)}, \ldots, \Theta^{(M)}\}$ exists, where $\Theta^{(m)}$ represents the set of all parameters needed to identify the m'th primitive. Given an observed trajectory $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_T\}$, our goal is to recognize which of the primitives from $\mathcal{L}$ are present in $\mathbf{Y}$ and where the switching points between these primitives are. Furthermore, if the trajectory $\mathbf{Y}$ contains a thus far unknown primitive, we wish to learn the parameters $\Theta^{(M+1)}$ of this new primitive and update the library to include $\Theta^{(M+1)}$, $\mathcal{L}^{new} = \mathcal{L}^{old} \cup \Theta^{(M+1)}$. Finally, we tackle the segmentation problem in a sequential manner, so that we do not need to know the number N of motion primitives $\mathbf{Y} = [\mathbf{Y}^{(1)}, \mathbf{Y}^{(2)}, \ldots, \mathbf{Y}^{(N)}]$ that are present in $\mathbf{Y}$. Thus, segmentation proceeds in the following steps:

Step 1: Candidate Points for Segmentation: The first step of our algorithm is to identify potential segmentation points of trajectory $\mathbf{Y}$. Given $\mathbf{Y}$, the velocity profile $\mathbf{V}$, and acceleration profile $\dot{\mathbf{V}}$ can be computed. We can identify possible segmentation points by finding minima in the velocity and/or acceleration profile, which is motivated by studies of human movement [11]. Here we look at the combination of $\mathbf{V}$ and $\dot{\mathbf{V}}$, $h(t) = v(t)^2 + \dot{v}(t)^2$. We then identify all local minima of $h(t)$ as hypothetical segmentation points $\mathcal{H} = \{h_0, h_1, \ldots, h_K\}$, where K is the number of local minima in $h(t)$.

Step 2: Matching: Given $\mathcal{H}$ we attempt to sequentially recognize motion primitives from the library. We assume that Y_1 corresponds to the starting point of the first segment. Then for each $h \in \mathcal{H}$ we find the parameter set $\Theta^{(m)}$ in $\mathcal{L}$ that has the highest likelihood of producing $Y_{1:h}$. Thus, we obtain K likelihood values l_k, one for each h_k. Is the largest likelihood value above a threshold l_{min}, $\max_k(l_k) > l_{min}$, then the corresponding h_k is chosen as segmentation point. This point then also forms the new starting point of the following segment and we can repeat this matching procedure. The difficulty of this step lies in the fact that we cannot assume that one of the $h \in \mathcal{H}$ corresponds to the actual end point of the current segment. Rather, we have to assume that the current primitive has been smoothed together with the following primitive, a phenomenon addressed as co-articulation in the behavioral literature [13]. Thus we need a recognition algorithm and a movement representation which are suitable for recognition of partially observed motions, an issue that we address in the next Section.

Step 3: New Segment: If no confident match can be found, we have to find the segmentation point that allows creating a new primitive with the highest statistical confidence, and update the library. The difficulties and details of this part are described in Section 4. Finally *Step 2* or *Step 3* are repeated until the end of the trajectory has been reached.

Before going into algorithmic details of the movement recognition, we briefly explain a movement representation that facilitates this segmentation procedure.

3 Linear Dynamic System Formulation of Dynamic Movement Primitives

Dynamic Movement Primitives (DMPs) were suggested in [14] as a representation to encode movement trajectories in terms of the attractor dynamics of weakly nonlinear differential equations. For instance, for a 1 DOF system, the equations are:

$$\begin{aligned} \dot{z}/\tau &= \alpha_z(\beta_z(g-p)-z) + sf(x) \\ \dot{p}/\tau &= z \\ \dot{x}/\tau &= -\alpha_x x \end{aligned} \tag{1}$$

with the nonlinear function

$$f(x) = \frac{\sum_{i=1}^{N} \psi_i w_i x}{\sum_{i=1}^{N} \psi_i}$$

where

$$\psi_i = \exp\big(-h_i(x-c_i)^2\big)$$

are exponential basis functions with center c_i and bandwidth h_i, and

$$s = \frac{g-p_0}{g_{fit}-p_{0,fit}} = \frac{g-p_0}{\Delta g}$$

is a scaling factor.

These equations encode a point attractor at the movement target g, and $p, \dot{p}, \ddot{p} = \dot{z}$ are position, velocity, and acceleration of the movement trajectory. In general, it is assumed that the movement duration τ and goal position g are known. Thus, given τ and g, the DMP is parameterized by weights $\mathbf{w} = (w_1, ..., w_N)^T$ of a function approximator f which can be trained to represent the shape of any smooth movement. During this fitting process, the scaling variable s is set to one, and the value of Δg, the difference between goal g and start state p_0, is stored as a constant for the DMP. Details can be found in [14]. It should be noted that DMPs have useful invariance properties from dynamic systems theory, i.e., the parameters $\mathbf{w}$ remain the same if a movement is scaled in space or time. Thus, DMPs represent a family of movements, not just one instance, and primitive libraries can be rather compact.

In our problem of movement segmentation and movement recognition, the parameter roles are reversed. We are given a library of DMPs $\mathcal{L} = \{\Theta^{(1)}, \ldots, \Theta^{(m)}, \ldots, \Theta^{(M)}\}$, where M is the number of movement primitives in the library and $\Theta^{(m)} = \{\mathbf{w}^{(m)}, \Delta g^{(m)}\}$. Assuming we know which primitive has generated a partially observed motion $\mathbf{Y}$, we can plug the corresponding $\mathbf{w}$ and Δg in (1), but are left wondering about which value to use for τ and g. We would like to determine these values such that the distance between the trajectory $\mathbf{P}$ produced by (1) and the observed trajectory $\mathbf{Y}$ is minimal.

To accomplish this goal, we realize that the original DMP can be reformulated in form of a linear dynamical system, with τ and g as the key system parameters. As a result, the estimation of τ, g becomes a system identification problem, and the similarity measure between $\mathbf{Y}$ and $\mathbf{P}$ is given through the likelihood $p(\mathbf{Y}|\tau, g)$.

3.1 Reformulation

The DMP equations can be discretized using Euler discretization with time step Δt, resulting in

$$\begin{aligned} x_t &= -\alpha_x\, x_{t-1}\tau\, \Delta t + x_{t-1} \\ z_t &= (\alpha_z(\beta_z(g-p_{t-1}) - z_{t-1}) + sf(x_{t-1}))\tau\, \Delta t + z_{t-1} \\ p_t &= z_{t-1}\tau\, \Delta t + p_{t-1} \end{aligned}$$

Next, we formulate the discrete time DMP as a linear dynamical system with inputs and Gaussian noise. Let $\mathbf{s}_t = \begin{pmatrix} z_t & p_t \end{pmatrix}^T$ be the (hidden) state of the primitive and y_t the observed trajectory point at time step t. We can write the stochastic DMP as

$$\begin{aligned} \mathbf{s}_t &= \mathbf{A}_1\mathbf{s}_{t-1} + \mathbf{A}_2\mathbf{s}_{t-1}\tau + \mathbf{B}\ \tau\ u_{t-1} + \epsilon \\ y_t &= \mathbf{C}\ \mathbf{s}_t + v \end{aligned}$$

where $\epsilon \sim N(0, \mathbf{Q})$ and $v \sim N(0, \mathbf{R})$. The state transition matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ are defined as

$$\mathbf{A}_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \mathbf{A}_2 = \begin{pmatrix} -\alpha_z\, \Delta t & -\alpha_z\beta_z\, \Delta t \\ \Delta t & 0 \end{pmatrix}.$$

The control input matrix $\mathbf{B}$ and the observation matrix $\mathbf{C}$ are set to

$$\mathbf{B} = \begin{pmatrix} \Delta t \\ 0 \end{pmatrix},\ \mathbf{C} = \begin{pmatrix} 0 & 1 \end{pmatrix}$$

and the control input u_t is computed as

$$u_t = \alpha_z\beta_z g + sf(x_t).$$

Note that the phase variable x is not part of the state s_t and only influences the input u_t.

3.2 Parameter Estimation

The parameters of the stochastic DMP formulation are given by

$$\theta_{all} = \{\mathbf{w}, \Delta g, \tau, g, \mathbf{A}_1, \mathbf{A}_2, \mathbf{B}, \mathbf{C}, \mathbf{Q}, \mathbf{R}\}.$$

However, the state transition, control input, and observation matrices are fixed and do not depend on the problem setting. Furthermore, we assume that the weights $\mathbf{w}$ and the value of Δg are known for primitives in the library. Thus the open parameters that can influence the likelihood of observing a trajectory $\mathbf{Y}$ are

$$\theta = \{\tau, g, \mathbf{Q}, \mathbf{R}\}$$

The goal is to estimate these parameters given an observed trajectory $\mathbf{Y} = \{y_t\}_{t=1}^T$ using maximum likelihood estimation. Because the model has hidden variables $\mathbf{S} = \{\mathbf{s}_t\}_{t=1}^T$ we use an Expectation Maximization algorithm to estimate these variables.

The complete data log likelihood is given by:

$$\begin{aligned}\ln p(\mathbf{Y}, \mathbf{S}|\tau, g) = \ln p(\mathbf{s}_1) + \\ \sum_{t=2}^{T} \ln p(\mathbf{s}_t|\mathbf{s}_{t-1}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{B}, u_{t-1}, \mathbf{Q}, \tau, g) + \\ \sum_{t=1}^{T} \ln p(y_t|\mathbf{s}_t, \mathbf{C}, \mathbf{R})\end{aligned}$$

Taking the expectation of the complete-data log likelihood with respect to the posterior $p(\mathbf{S}|\mathbf{Y}, \theta^{old})$ defines the function

$$Q(\theta, \theta^{old}) = \mathbb{E}_{\mathbf{S}|\theta^{old}}\big[\ln p(\mathbf{Y}, \mathbf{S}|\theta)\big],$$

which we want to maximize with respect to θ. The maximum likelihood solution of all parameters in θ can be found analytically. Taking the derivative of $Q(\theta, \theta^{old})$ with respect to τ[1] and solving for τ, yields

$$\tau = \frac{\Sigma_1}{\Sigma_2}$$

[1]Note that the phase variable x_t is not part of state s_t although x_t depends on τ. However, one can show that the inclusion of x_t in s_t with the assumption of no noise on x_t, leads to the same update equation for τ.

Table 1: Step 2: Matching

- **Given**
  - set of possible segmentation points $\mathcal{H}$
  - likelihood threshold p_{min}
  - first data point Y_1 of new segment
  - library of primitives $\Theta = \{\Theta^{(m)}\}_{m=1}^M$
- **for each** $h_k \in \mathcal{H}$
  - **for each primitive in library**
    - execute EM on $\mathbf{Y}_{1:h_k}$ to optimize parameters $\tau^{(m)}, g^{(m)}$ and compute log likelihood $l_k^{(m)} = \ln p(\mathbf{Y}_{1:h_k}|\theta^{(m)})$
  - find the most likely primitive for point h_k : $m_{ml,k} = \arg\max_m l_k^{(m)}$
- find most likely segmentation point h_{k^*}, $k^* = \arg\max_k l_k^{(m_{ml})}$
- if $l_{k^*}^{(m_{ml})} > l_{min}$, then match : m_{ml,k^*} found, otherwise execute *Step 3*.

where Σ_1 and Σ_2 are given through

$$\begin{aligned}\Sigma_1 = \sum_{t=2}^{T} \big\{ & Tr(\mathbf{A}_2^T \mathbb{E}[\mathbf{s}_t \mathbf{s}_{t-1}^T]) + \mathbb{E}\big[\mathbf{s}_t^T\big]\mathbf{B}u_{t-1} - \\ & Tr(\mathbb{E}\big[\mathbf{s}_{t-1}\mathbf{s}_{t-1}^T\big]\mathbf{A}_1^T\mathbf{A}_2) - \mathbb{E}\big[\mathbf{s}_{t-1}\big]^T\mathbf{A}_1^T\mathbf{B}u_{t-1}\big\} \\ \Sigma_2 = \sum_{t=2}^{T} \big\{ & Tr(\mathbb{E}\big[\mathbf{s}_{t-1}\mathbf{s}_{t-1}^T\big]\mathbf{A}_2^T\mathbf{A}_2) + \\ & 2\mathbb{E}\big[\mathbf{s}_{t-1}^T\big]\mathbf{A}_2^T\mathbf{B}u_{t-1} + u_{t-1}^T\mathbf{B}^T\mathbf{B}u_{t-1}\big\}.\end{aligned}$$

Optimizing for g results in the update $g = \frac{1}{\tau}(\Sigma_4)^{-1}\Sigma_3$, with

$$\begin{aligned}\Sigma_3 &= \sum_{t=2}^{T} \hat{u}_{t-1}^T \mathbf{B}^T \mathbf{B} \hat{u}_{t-1}, \\ \Sigma_4 &= \sum_{t=2}^{T} \hat{u}_{t-1}^T \mathbf{B}^T (\mathbf{s}_t - \mathbf{A}_1\mathbf{s}_{t-1} - \mathbf{A}_2\mathbf{s}_{t-1}\tau) \\ \hat{u}_t &= \frac{f(x_t)}{\Delta g} + \alpha_z\beta_z\end{aligned}$$

To compute the maximum likelihood estimation of θ, the following expectations are needed:

$$\begin{aligned}\mathbb{E}\big[\mathbf{s}_{t-1}\big] &= \hat{\mathbf{s}}_{t-1} \\ \mathbb{E}\big[\mathbf{s}_{t-1}\mathbf{s}_{t-1}^T\big] &= cov(\mathbf{s}_{t-1}, \mathbf{s}_{t-1}) + \hat{\mathbf{s}}_{t-1}\hat{\mathbf{s}}_{t-1}^T \\ \mathbb{E}\big[\mathbf{s}_t\mathbf{s}_{t-1}^T\big] &= cov(\mathbf{s}_t, \mathbf{s}_{t-1}) + \hat{\mathbf{s}}_t\hat{\mathbf{s}}_{t-1}^T.\end{aligned}$$

The estimates of the state and covariance matrices are calculated through Kalman smoothing. Given the ability to optimize a partially observed trajectory for it's duration τ and goal position g, we can realize the matching step as described in Table 1.

4 Updating the Library

In the case that none of the primitives in the library can satisfactorily fit any of the potential segments, we have to assume that we are observing a new primitive. Thus, it is necessary to determine the parameters of this new primitive and include them in the library. Given a trajectory, the DMP formulation in (1) allows computing the weights $\mathbf{w}$ with weighted linear regression. In order to do so, two problems need to be addressed. First, it is not clear which of the possible segmentation points to use. Second, when trying to compute the maximum likelihood estimate of the weights $\mathbf{w}$, the duration τ and goal position g need to be known – however it can not be assumed that the segmentation point h is the true goal position of the underlying primitive.

4.1 Trajectory Completion

The second issue is readily taken care of by assuming that the segment point is close to the trajectory of a trajectory, and thus it is plausible to assume that we are close to the true goal of the current primitive. As a result the phase variable x should have a small value[2]. Note, the value of the phase variable $x(t)$ at time step t depends on the duration τ. More specifically, it holds $x(t) = exp(-\alpha_x \tau\ \Delta t\ t)$. Thus, by assuming that the switch between current primitive and the next one happens at a small value of $x(t) = \eta$, we can solve $\eta = exp(-\alpha_x \tau\ \Delta t\ t)$ for τ, resulting in the estimate of the true duration $\hat{\tau} = -\frac{log(\eta)}{\alpha_x \Delta t\ t}$. Given this estimate, we can infer at what time step T the trajectory should converge to the goal. While there are infinitely many possibilities to fill the missing data points, we know that DMPs are designed such that the velocity as well as the acceleration profile smoothly converge towards zero at the end of a movement. Hence, we complete the velocity profile $V = \dot{Y}$ using spline interpolation with boundary conditions $v(T) = 0, \dot{v}(T) = 0$. The completed velocity V profile is integrated to obtain the completed trajectory Y, and with that the estimate $\hat{g}$ of the true goal position. Given the estimates $\hat{\tau}$ and $\hat{g}$, the weights $\mathbf{w}$ can be computed as follows: Evaluate the values of the phase variable $x(t), t = 1 \dots T$, from (1) compute the regression target $f_{target}(x(t))$ using the observed and completed $Y, V, \dot{V}$, then estimate the weights $\mathbf{w}$ such that the error $J = \sum_{t=1}^{T} (f_{target}(x(t)) - f(x(t)))^2$ is minimized – more information on how to fit DMPs to data can also be found in [14].

[2] x is 1 at the start of a movement and decays monotonically to 0, which is approximately reached after the movement duration τ

4.2 Selection of Segmentation Point

It now remains the question which segmentation point of $\mathcal{H} = \{h_0, h_1, \dots, h_K\}$ to choose. In theory, we can perform the procedure of the previous paragraph for each $h \in \mathcal{H}$, resulting in a set of possible weight parameters $\{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_K\}$. However, if one were to choose the $\mathbf{w}_k$ that results in the smallest normalized error $\frac{J_k}{|Y_{1:h_k}|}$, where $|Y_{1:h_k}|$ is the number of data points for the k'th segmentation point, one would end up preferring shorter trajectories. This is similar to the model selection problem in typical regression settings, where the use of more parameters leads to better fitting results and thus higher likelihood values. Just that in our setting, the number of parameters is fixed, but the length of the trajectory varies, and with that the number of data points for fitting varies as well. The shorter a trajectory, usually the better it can be fit using a fixed number of basis functions. Similar to the typical regression model selection problem, we can turn to a Bayesian treatment of linear regression to find the optimal segmentation point $h \in \mathcal{H}$ for a fixed number of basis functions.

In the following, let $\mathbf{Y}$ denote the target values of our regression problem and X the input values. Without loss of generality, assume that the segment starts at Y_1. Then the normalized log model evidence, as a function of segmentation point h, is given by $E(h) = \frac{\ln p(\mathbf{Y}_{1:h})}{h}$. The joint probability is given through $p(\mathbf{Y}, X, \mathbf{w}, \beta; \alpha) = p(\mathbf{Y}|X, \mathbf{w}, \beta; \alpha) p(\mathbf{w}, \beta; \alpha)$, where it is assumed that the value of the hyperparameter α is known. Furthermore, the prior distribution over the weights $\mathbf{w}$ is modeled through an isotropic Gaussian with zero-mean, and the prior distribution over precision parameter β is given through a Gamma distribution, resulting $p(\mathbf{w}, \beta) = \mathcal{N}(w|0, \beta^{-1}\mathbf{S}_0)\, Gam(\beta|a, b)$, with $S_0 = \alpha^{-1}\mathbf{I}$. The model evidence $p(\mathbf{Y}_{1:h})$ is obtained by integrating out the parameters $\mathbf{w}, \beta$

$$p(\mathbf{Y}_{1:h}|\alpha) = \int \int p(\mathbf{Y}_{1:h}|X, \mathbf{w}, \beta; \alpha) p(\mathbf{w}, \beta; \alpha) d\mathbf{w}\ d\beta.$$

Given α, the integral can be solved analytically, resulting in a model evidence value for each plausible segmentation point. To incorporate the fact that we do not have any useful prior information for the weights $\mathbf{w}$, we choose a very small value for α which leads to a broad uninformative prior distribution over the weights. Finally, we need to consider that part of the log model evidence is the sum of errors of all data points. Thus, the more data points the larger the absolute error, and the log model evidence $\ln p(\mathbf{Y}_{1:h_k})$ decreases. Because of that we normalize the evidence with the number of data points h_k in a trajectory piece $Y_{1:h_k}$ to obtain the average *per data point log model evidence*. The segmentation point $h \in \mathcal{H}$ is then chosen

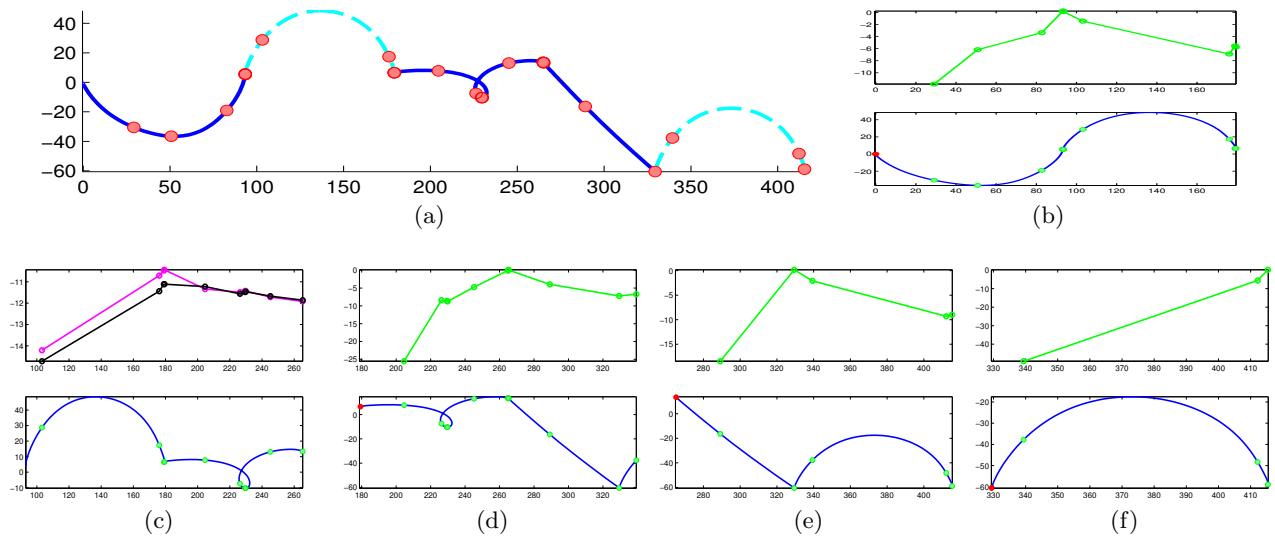

Figure 1: (a) Observed $2D$ trajectory with all plausible segmentation points $h \in \mathcal{H}$ shown as red dots; the segments are semi-circle, semi-circle, squiggle, line, semi-circle in this example; (b)-(f) the bottom plot shows the current partial trajectory considered for segmentation. The red dot indicates the starting point of current segment, the green dots indicate the possible segmentation points. The plot above, depicts the normalized log likelihood of the best match in $\mathcal{L}$ for each plausible switching point. In (c), no match in $\mathcal{L}$ can be found, thus the normalized log model evidence (of both dimensions) is used to determine the segmentation point.

as

$$h = \arg\max_k E(h_k) = \arg\max_k \frac{\ln p(\mathbf{Y}_{1:h_k})}{h_k}$$

5 Experiments

In this section we first present some results on some synthetic drawing data and then on some human movement data. The experiments consist of a training and testing phase. In the training phase, recorded motions that correspond to predefined primitives were used to create the library. First the average weights of each primitive were computed using weighted linear regression. Then, given these averaged weights, and the true values for τ and g the observation noise matrices are learned using the EM algorithm, similar to Section 3.2, only τ, g are fixed, and R is updated. Also, we initialize the values $\tau^{(m)}$ and $g^{(m)}$ to the mean duration and goal position of the training instances for primitive m. For both data sets, each movement primitive is parametrized by 20 weights.

5.1 Baseline Evaluation

In order to perform a baseline evaluation, we also implemented a recognition and segmentation algorithm based on Dynamic Time Warping (DTW). Here, the library was created by randomly picking one of the recorded training instances of each primitive m and storing that trajectory as primitive m. The segmentation procedure is essentially the same as described in Section 2. Possible segmentation points $\mathcal{H}$ are found as described in *Step 1*. Then for each possible segmentation point $h_k \in \mathcal{H}$, the trajectory piece $Y_{1:h_k}$ is matched against each primitive m in the library using Dynamic Time Warping resulting in a distance value $d_{k,m}$. This distance value is the normalized warping distance and serves as deciding factor: The combination of h_k and primitive m that results in the smallest distance $d_{k,m}$ is chosen to provide a segmentation point and recognition result.

Note, that while we can also define some threshold on $d_{k,m}$ to identify situations when unknown primitives are encountered, it is not clear how to choose the segmentation point when no primitive in the library fits. There are no constraints that could help identify a suitable choice $h_k \in \mathcal{H}$, making a purely sequential segmentation strategy when the library is *incomplete* unsuitable. Thus, we have chosen to test our proposed algorithm with a *complete* and *incomplete* library. For segmentation with a complete library, the update step (*Step 3*) is not needed, and we can easily compare to the DTW approach.

5.2 2D-Data

To demonstrate the functionality of our algorithm, we present results on some toy data. Using a digitizing tablet, we recorded 3 simple shapes - semi-circle, line and squiggle - in different configurations, resulting in a library of 10 primitives. For each of these prim-

	# of correctly segmented sequences	# of correct segmentation points	# of correctly classified segments
baseline DTW - *complete* library	10 (of 20)	80 (of 100)	82 (of 100)
stochastic DMP - *complete* library	19 (of 20)	98 (of 100)	98 (of 100)
stochastic DMP - *incomplete* library	17 (of 20)	96 (of 100)	96 (of 100)

Table 2: Segmentation Results

itives 10 instances are used to train the parameters $\Theta^{(m)}$. For testing purposes, another 5 instances were retained to generate random primitive sequences by first randomly selecting 5 primitives (duplication possible), then randomly selecting values for duration and goal position of each segment, and finally concatenating them to obtain a sequence. These sequences were smoothed using a butterworth filter to assure that no obvious segmentation points appear. For the segmentation experiment with an *incomplete* library one of the 5 primitives in the testing sequence was randomly chosen to be deleted from the library. An example of this can be seen in Figure 1. The cyan dotted line indicate the trajectory parts of the primitive that was deleted from the library. We now start with the first data point as start position and try to sequentially determine the segments in $\mathbf{Y}$. The first segment is readily determined as can be seen in Figure 1b. For efficiency reasons, only segmentation points $h \in \mathcal{H}$ are considered that would produce a trajectory length within the interval of [30; 300], inspired by the fact that most human movement durations are in the range of 0.3 to 3 seconds. Clearly, h_4 has the highest likelihood and is thus chosen as segmentation point, and thus forms the new starting point of the next segment. We also know which primitive in the library generated this highest likelihood, such that we segmented and recognized the first part of the trajectory. For the next segment, none of the possible segmentation points produces a normalized likelihood value above our threshold $l_{min} = -3.0$, thus no confident match could be found. Hence, the model evidence obtained through Bayesian linear regression is used to chose one of the possible switching points, and the parameters of this new primitive are learned and included in the library. Segments 3 and 4 can again be confidently matched against a primitive in $\mathcal{L}$. The last segment can now be recognized as the newest member of $\mathcal{L}$.

We have performed the segmentation first with a *complete* and then with an *incomplete* library, for 20 randomly concatenated sequences, each sequence consisting of 5 segments. Additionally, we have also tested the DTW approach on these sequences for the *complete* library case. The results are shown in Table 2. The first column shows, how many sequences were correctly segmented, with all primitives being correctly recognized. In the second column we see how many of the 100 segmentation points were correctly chosen, and in column three how many of the 100 segments correctly recognized. Note, that these two numbers need not be the same, as segments can still be correctly classified even when their segmentation point has been chosen sub optimally, or the other way around.

While the DTW approach only manages to segment half of the sequences flawlessly, we can see that the stochastic DMP approach does perform very well when the library is complete as well as when it is incomplete. This indicates, that the stochastic DMP approach is suitable for recognition and segmentation. Furthermore, the update of the library seems to perform very well. Compared to the *complete* library experiment only 2 more segments were wrongly classified and 2 more segmentation points wrongly chosen. Also, in 5 of the 20 sequences, the primitive that was deleted from the library appeared two or more times in the sequence, as indicated in Figure 1. For all these sequences the library was updated when the new primitive was encountered first, and then correctly classified as this new primitive later in the sequence.

5.3 3D-Data

We also tested our approach on some human movement data. Specifically, we recorded the movement sequence of 'moving to pour' (A) milk into coffee, 'putting back' (B) milk, 'reaching for' (C) sugar, 'moving to pour' (A) sugar into coffee, 'putting back' (B) sugar, 'reaching for spoon' (D), 'putting spoon in cup' (E), 'stir' (F) three times, 'taking out spoon' (G) and 'putting back' (B) spoon. This movement sequence and the corresponding segmentation into the underlying primitives can be seen in Figure 2a, where each primitive is represented by a capital letter from $A - G$. We obtained the $3D$ endeffector trajectory of this movement sequence using the 3D Guidance trakStar system. The data was recorded with a sampling frequency of 60Hz and the continuous movement sequences have a duration of 15 to 20 seconds. Furthermore, we recorded 10 instances of motions representing each of the 7 primitives iterated above, which we used to create a library. Using this library we then first segment the continuous movement sequence with the complete library using the DTW approach and the stochastic DMP approach. The result of this experi-

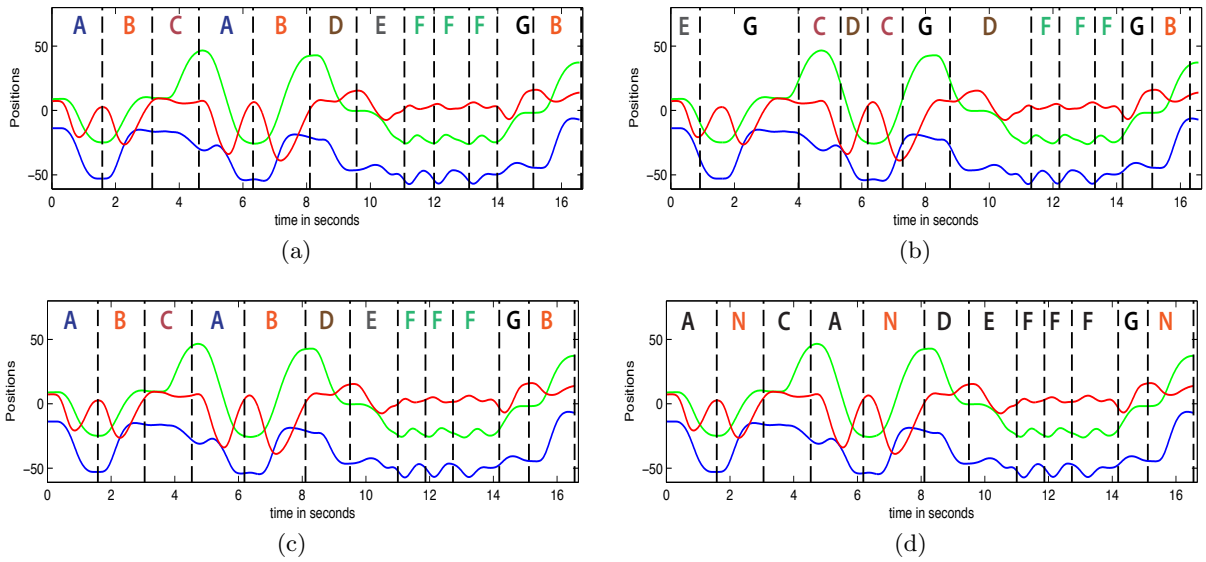

Figure 2: (a) Ground truth (b) Segmentation results using DTW (c) Segmentation results using stochastic DMPs (d) Segmentation results with stochastic DMPs and missing primitive B. In all plots, the blue, green and red curves correspond to x,y and z trajectories, respectively.

ment can be seen in Figure 2b and 2c, respectively. Clearly, the DTW approach is outperformed by our proposed approach in both recognition and segmentation precision. Our approach correctly recognizes all segments, and except one correctly identifies all segmentation points. One segmentation point was chosen to greedily - second stir motion (primitive F) – however in this case the parsing of the rest of the sequence is not affected. Using our framework, the same sequence is segmented again after primitive B has been deleted from the library, see Figure 2d. Now, when the unknown primitive is encountered, it is learned and stored in library as primitive N and later on correctly recognized. As before, all the known primitives were correctly identified and the same segmentation points were chosen.

6 Discussion and Conclusion

We have presented a sequential movement segmentation and recognition framework based on a reformulated version of Dynamic Movement Primitives. As a result, observed trajectories can be segmented into primitives and reproduced again with different durations, goals or starting positions. Furthermore, we have shown that we can create new primitives in an unsupervised fashion, a process that is guided by the DMP design and leans on the Bayesian model evidence framework.

For any given sequence of primitives, we expect our algorithm to identify the known primitives (library), and learn new primitives if no match in the library can be found. Thus, if the primitives in the library are meaningful, then so are the segments that were identified as these primitives. As for newly learned primitives, our algorithm only considers segments that are meaningful in the Dynamic Movement Primitive (DMP) framework, as we only consider end points with small velocity/acceleration and Bayesian Linear Regression to choose the point that produces the largest model evidence given a fixed number of basis functions. The strong bias from the DMP framework ensures meaningful primitives in the sense of what we know from human arm movements.

Thus, the segmentation results mostly depend on the library and with that it is in the users power to obtain the segmentation results he/she prefers. If the user wants a fine/coarse segmentation, the library should consist of shorter/longer primitives, respectively. For newly learned primitives, the user can define the number of basis functions that should be used. The more basis functions the more complicated (longer) a new primitive can be.

A limitation of our approach, as described in this paper, is the fact that if one of the segmentation points has been chosen incorrectly, the segmentation of the rest of the trajectory can be very much affected. This can happen especially when there are multiple good choices either from the matching against the library step or the bayesian segmentation point selection step. This problem can be easily alleviated by using a Dynamic Programming approach to perform a backward search. In this case, outputs of the matching step or the new primitive creation step should be regarded as rankings of segmentation points.

References

[1] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233–242, 1999, clmc.

[2] T. Inamura, T. Iwaki, H. Tanie, and Y. Nakamura, "Embodied symbol emergence based on mimesis theory," *International Journal of Robotics Research*, vol. 23, no. 4-5, pp. 363–377, 2004.

[3] O. Jenkins and M. Mataric, "Performance-derived behavior vocabularies: Data-driven acqusition of skills from motion," *International Journal of Humanoid Robotics*, vol. 1, no. 2, pp. 237–288, Jun 2004.

[4] F. Zhou, F. Torre, and J. Hodgins, "Aligned cluster analysis for temporal segmentation of human motion," in *Automatic Face & Gesture Recognition*, 2008.

[5] M. Alvarez, J. Peters, B. Schölkopf, and N. Lawrence, "Switched latent force models for movement segmentation," *Advances in Neural Information Processing Systems*, vol. 23, pp. 55–63, 2010.

[6] B. Williams, M. Toussaint, and A. Storkey, "Extracting motion primitives from natural handwriting data," *Int. Conf. on Artificial Neural Networks*, pp. 634–643, 2006.

[7] T. Kim, G. Shakhnarovich, and R. Urtasun, "Sparse coding for learning interpretable spatio-temporal primitives," *Advances in Neural Information Processing Systems*, 2010.

[8] A. Bobick and Y. Ivanov, "Action recognition using probabilistic parsing," in *Computer Vision and Pattern Recognition*, 1998.

[9] F. Lv and R. Nevatia, "Recognition and segmentation of 3-d human action using hmm and multi-class adaboost," in *European Conference on Computer Vision*, 2006, pp. 359–372.

[10] J. Rittscher and A. Blake, "Classification of human body motion," in *International Conference on Computer Vision*, 1999.

[11] D. Weinland, R. Ronfard, and E. Boyer, "A survey of vision-based methods for action representation, segmentation and recognition," *Computer Vision and Image Understanding*, 2011.

[12] S. Chiappa and J. Peters, "Movement extraction by detecting dynamics switches and repetitions," *Advances in Neural Information Processing Systems*, 2010.

[13] R. Sosnik, T. Flash, B. Hauptmann, and A. Karni, "The acquisition and implementation of the smoothness maximization motion strategy is dependent on spatial accuracy demands," *EXPERIMENTAL BRAIN RESEARCH*, vol. 176, no. 2, pp. 311–331, 2007.

[14] A. Ijspeert, J. Nakanishi, and S. Schaal, "Learning attractor landscapes for learning motor primitives," *Advances in neural information processing systems*, pp. 1547–1554, 2003.