Temporal Segmentation of Pair-Wise Interaction Phases in Sequential Manipulation Demonstrations

Andrea Baisero[1] Yoan Mollard[2] Manuel Lopes[2] Marc Toussaint[1] Ingo Lütkebohle[1]

***Abstract*— We consider the problem of learning from complex sequential demonstrations. We propose to analyze demonstrations in terms of the concurrent interaction phases which arise between pairs of involved bodies (hand-object and object-object). These interaction phases are the key to decompose a full demonstration into its atomic manipulation actions and to extract their respective consequences. In particular, one may assume that the goal of each interaction phase is to achieve specific geometric constraints between objects. This generalizes previous Learning from Demonstration approaches by considering not just the motion of the end-effector but also the relational properties of the objects' motion.**

We present a linear-chain Conditional Random Field model to detect the pair-wise interaction phases and extract the geometric constraints that are established in the environment, which represent a high-level task oriented description of the demonstrated manipulation. We test our system on single- and multi-agent demonstrations of assembly tasks, respectively of a wooden toolbox and a plastic chair.

I. INTRODUCTION

A major challenge in robotics is to provide intuitive ways for non-experts to instruct and work cooperatively with robotic systems. This is true in household domains, where non-expert consumers demand personalized functionalities, as well as in new industries, which are required to be much more flexible in their production lines.

Learning from Demonstration (LfD) is a teaching paradigm where a robotic system learns to perform new tasks by observing examples of respective demonstrations [5], [19]. While such approaches are intuitive for the user, they are limited in several aspects. With few exceptions, previous LfD approaches have focused on teaching single motions individually rather than complex sequential manipulations as a whole. Such type of low level teaching provides data almost exclusively for the motion primitives themselves, while instructing complex and sequential tasks seems more efficient and intuitive on a more abstract level.

We approach LfD under the assumption that sequential manipulation can be well understood as a set of potentially concurrent interaction phases, where the goal of each phase is to move some objects into geometric configurations which we call *constraints*—in essence, a constraint is the negation of some degree of freedom; e.g. the configuration of a chair's leg w.r.t. its seat is an example of *rigid* constraint.

Work supported by the 3rdHand project, funded by the European Union under the FP7 programme (FP7-ICT-2013-10610878).

[1]Machine Learning and Robotics Lab, University of Stuttgart, Germany. firstname.lastname@ipvs.uni-stuttgart.de

[2]Flowers Team, French Institute for Research in Computer Science and Automation (Inria), France. firstname.lastname@inria.fr

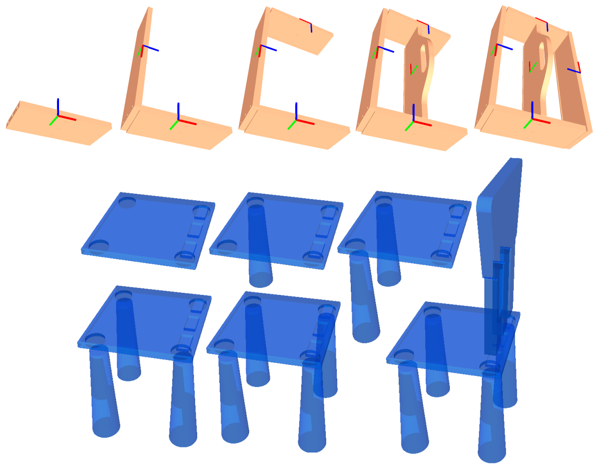

Fig. 1: High-level assembly task representation for a *toolbox* and a *chair* demonstration, each as a sequence of *rigid* constraints which are created between object parts. The figures represent actual output of the proposed model.

If there are multiple manipulators and objects, then multiple such manipulator-object (or object-object) interactions can concurrently be active. Under this view, instructing a robot to perform a sequential manipulation implies demonstrating which interactions should be performed, in which order, and what their desired outcomes are. An effective LfD algorithm should therefore be capable of detecting such interaction phases and extracting the respective constraints. This approach leads to a much more abstract understanding of the task, as opposed to directly modeling low-level motions.

We train a Conditional Random Field (CRF) to detect the interaction phases between any pair of moving bodies (manipulator-object or object-object). Roughly, the CRF learns to exploit features that indicate a temporarily rigid transformation between two moving bodies. The CRF automatically segments new manipulation demonstrations into the respective set of potentially concurrent interaction phases. In a second stage, we analyze the outcome of these interactions phases w.r.t. the geometric constraints that have been established by the interaction. This gives a task-space description of the goal of each interaction phase and, thereby, a goal-oriented analysis of the demonstration (Fig. 1).

For simplicity, we focus on detecting *rigid* constraints exclusively; however, we show that rigid constraints are sufficient both to detect pick-and-place manipulations and to describe furniture assembly tasks, which is the main setting of our work.

II. RELATED WORK

Learning from Demonstration algorithms [5] have mostly focused on forms of demonstration where any hierarchical or sequential aspect is explicitly described by the teacher. The notion of key-frame demonstration was introduced in [1], [2], where users are asked to provide key aspects in the form of the most important via-points, as opposed to full demonstrations. Recent research started to consider how a complex demonstration can be represented and decomposed in simpler parts autonomously.

Recent approaches to action recognition and object affordance classification have started to focus on human-object and object-object interactions. Aksoy et al. [3] propose a model-free approach to human activity modeling based on the generation of semantic scene graphs and symbolic event tables which represent the temporal evolution of the spatial relations between entity segments. Pieropan et al. [18] extend this approach by increasing the symbolic relation dictionary, taking into account more dynamic relations between entities, and using a variety of string kernels to perform classification on sequential data. Vafeias et al. [20] use a Hidden-state CRF to model human motion and object state changes jointly, this time for the sole purpose of activity recognition.

Niekum et al. presented in [17], [15] an integrated approach to segment manipulation demonstrations into actions, each represented by a Dynamic Motion Primitive (DMP). The segmentation of demonstrations is realized with a Beta Process autoregressive Hidden Markov Model (BP-AR-HMM) [22] which produces both the segmentation and the corresponding association between latent variable values and corresponding learned DMP. This approach is promising to identify segments of robot motion and compile these into DMPs. However, our aim is to identify the crucial interaction phases between manipulators and objects that correspond to direct manipulation. In [16], Niekum et al. extend previous work on changepoint detection and develop a new algorithm which is used to infer the dynamic changes in the articulation structure between objects in an assembly task.

A series of works [21], [7], [13], [4], [9] formulate integrated probabilistic models of sequential or superimposed motion primitives which, when fitted to data, imply a segmentation of motions. Again, the goal of these approaches is to extract specific motion primitives from data rather than more abstract representations of the occurred manipulations. Barbič et al. [6] provide a good discussion of traditional methods for segmenting motion capture data, including the detection of zero crossings of angular velocities [8], and their own PCA approach. Incremental creation of motor primitives has also been used to create a dictionary of full-body motions relying on zero-acceleration heuristics for segmentation [10]. Another approach relied on clustering to segment low-level sequences to learn motor primitives and grammar at an high level [12]. However, in all these approaches the problem setting focuses on the extraction of elemental motion primitives rather than analyzing pair-wise interaction phases between manipulators, tools and objects.

In conclusion, we are not aware of previous work that explicitly aimed at training a segmentation algorithm to identify atomic manipulations in the midst of a more complex demonstration as a prerequisite to describe such complex manipulation as a sequence of interactions which generate specific relations between the objects.

III. OVERVIEW OF CONDITIONAL RANDOM FIELDS

A CRF [11] models a conditional probability mass function as a normalized product of (typically log-linear) potentials,

$$p(\boldsymbol{x} \mid \boldsymbol{y}) = \mathcal{Z}(\boldsymbol{y})^{-1} \prod_k \psi_k(\boldsymbol{x}, \boldsymbol{y}) \,, \tag{1}$$

where the potentials may establish arbitrary correlations between any subset of the discrete latent variables $\boldsymbol{x}$ and the full set of observed variables $\boldsymbol{y}$. The partition function $\mathcal{Z}(\boldsymbol{y}) = \sum_{\tilde{x}} \prod_k \psi_k(\tilde{\boldsymbol{x}}, \boldsymbol{y})$ acts as the normalizing constant.

In a linear-chain CRF, the latent variables form a sequence which can be referenced by an index, and the potentials only couple directly adjacent latent variables. A linear-chain CRF describes the conditional probability mass function

$$p(\boldsymbol{x} \mid \boldsymbol{y}) = \mathcal{Z}(\boldsymbol{y})^{-1} \prod_{t=1}^{T} \psi_t(\boldsymbol{x}, \boldsymbol{y}) \,, \tag{2}$$

$$\psi_t(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(\phi_t(x_t, x_{t-1}, \boldsymbol{y})^\intercal \theta\right) , \tag{3}$$

where the *feature vector* ϕ_t is the concatenation of *transition features* $\phi_{\tau,t}$, which couple consecutive latent variables x_t and x_{t-1}[1]; and *state features* $\phi_{\sigma,t}$, which couple the individual latent variable x_t to the observation set $\boldsymbol{y}$.

A. Training

Model parameters θ are learned by running maximum-likelihood estimation on a set of labeled demonstrations. The neg-log likelihood of a linear-chain CRF is a convex function of θ; this makes it a relatively simple problem to optimize for which a variety of first- and second-order iterative algorithms already exist.

The neg-log likelihood of the model parameters associated with a set of labeled sequences $\mathcal{D} = \{(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})\}_{n=1}^{N}$ is

$$\mathcal{L}(\theta; \mathcal{D}) = \sum_{n=1}^{N} \mathcal{L}(\theta; \boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}) \,, \text{ where} \tag{4}$$

$$\begin{aligned} \mathcal{L}(\theta; \boldsymbol{x}, \boldsymbol{y}) &= -\log p(\boldsymbol{x} \mid \boldsymbol{y}) \\ &= \log \mathcal{Z}(\boldsymbol{y}) - \sum_{t=1}^{T} \phi_t(x_t, x_{t-1}, \boldsymbol{y})^\intercal \theta \,, \end{aligned} \tag{5}$$

[1]As a detail we mention that, because there is no latent variable x_0 in the model, the feature vector ϕ_1 needs special consideration: For simplicity of notation, all features that depend on x_0 are defined as identically zero.

whereas the respective gradient is

$$\nabla\mathcal{L}(\theta;\mathcal{D}) = \sum_{n=1}^{N} \nabla\mathcal{L}(\theta;\boldsymbol{x}^{(n)},\boldsymbol{y}^{(n)})\,,\text{ where} \tag{6}$$

$$\begin{aligned}\nabla\mathcal{L}(\theta;\boldsymbol{x},\boldsymbol{y}) = &\sum_{t=1}^{T}\sum_{\tilde{x}_t,\tilde{x}_{t-1}} \phi_t(\tilde{x}_t,\tilde{x}_{t-1},\boldsymbol{y})\; p_t(\tilde{x}_t,\tilde{x}_{t-1}\,|\,\boldsymbol{y}) \\ &- \sum_{t=1}^{T}\phi_t(x_t,x_{t-1},\boldsymbol{y})\,. \end{aligned}\tag{7}$$

The values of the partition function $\mathcal{Z}(\boldsymbol{y})$ and of the joint conditional probabilities $p_t(\tilde{x}_t,\tilde{x}_{t-1}\,|\,\boldsymbol{y})$, which also depend on θ, can be efficiently computed using the forward-backward algorithm, the details of which we leave out.

IV. DETECTION OF INTERACTION PHASES

In our application, an object is characterized solely by its trajectory—the pair $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_\pi,\boldsymbol{\alpha}_\rho)$, where $\boldsymbol{\alpha}_\pi = (\alpha_{\pi,t})_{t=1}^{T}$ is the trajectory of positions $\alpha_{\pi,t} \in$ T(3) and $\boldsymbol{\alpha}_\rho = (\alpha_{\rho,t})_{t=1}^{T}$ is the trajectory of orientations $\alpha_{\rho,t} \in$ SO(3). For notational simplicity, we use $\boldsymbol{\alpha}_\chi$ as a stand-in for either $\boldsymbol{\alpha}_\pi$ or $\boldsymbol{\alpha}_\rho$. We note that position vectors and quaternions are treated, for the purpose of feature computation, as elements of $\mathbb{R}^3$ and $\mathbb{R}^4$ respectively.

Given two trajectories $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, we model the interaction between the respective objects A and B with a linear-chain CRF. We define the latent variables $\boldsymbol{x}$ as a sequence of binary variables which represent the interaction between the objects at each time step, and the observation sequence as the observed trajectories of the two objects throughout the whole demonstration $\boldsymbol{y} = (\boldsymbol{\alpha},\boldsymbol{\beta})$. Additionally, we define $\boldsymbol{\delta}$ as the relative trajectory of B's frame w.r.t. A's frame:

$$\delta_{\pi,t} = \alpha_{\rho,t}^{-1}(\beta_{\pi,t}-\alpha_{\pi,t})\,, \tag{8}$$

$$\delta_{\rho,t} = \alpha_{\rho,t}^{-1}\beta_{\rho,t}\,. \tag{9}$$

A. Transition Features

As is typical for linear-chain CRFs, the transition features $\phi_{\tau,t}$ indicate the discrete latent state transitions and do not depend on the observation sequence $\boldsymbol{y}$ (although in principle they could):

$$\phi_{\tau,t}(x_t,x_{t-1},\boldsymbol{y}) = \begin{pmatrix} \mathbb{I}[x_t=0]\,\mathbb{I}[x_{t-1}=0] \\ \mathbb{I}[x_t=0]\,\mathbb{I}[x_{t-1}=1] \\ \mathbb{I}[x_t=1]\,\mathbb{I}[x_{t-1}=0] \\ \mathbb{I}[x_t=1]\,\mathbb{I}[x_{t-1}=1] \end{pmatrix}\,, \tag{10}$$

where $\mathbb{I}$ is the indicator function.

B. State Features

The type of interaction we are trying to detect is one where the same non-identical transformation appears to describe the motion of both objects, i.e., one where objects move both individually and together as a rigid body.

Functions which *quantify* the amount of motion in a sequence of transformations—which may represent either the motion of an individual object, or the relative motion between objects—are thus supposedly effective candidates to compute discriminative features for this task. We also note that, because position and orientation represent ontologically different entities, we will compute the features for transitional and rotational movements separately—albeit using the same functions.

Instantaneous linear and angular speeds seem straightforward choices for such features. However, instantaneous speeds may vary wildly between adjacent time steps, and exhibit very high magnitudes even in situations where the actual movement is negligible, thus resulting in noisy features and degraded performance. Rather, we define a discrete window length ω and compute two types of features from the ω most recent positions and rotations.

1) Variance-Based Features: The first proposed feature type computes a pair of scalar variance measures of the multi-dimensional positional and rotational motion sequences within the defined window of time:

$$f_t(\boldsymbol{\alpha}) = (f_t(\boldsymbol{\alpha}_\pi), f_t(\boldsymbol{\alpha}_\rho))^\intercal\,,\text{ where} \tag{11}$$

$$f_t(\boldsymbol{\alpha}_\chi) = \omega^{-1}\sum_{t'=t-\omega+1}^{t} \|\alpha_{\chi,t'} - \mu_t(\boldsymbol{\alpha}_\chi)\|^2 \text{ and} \tag{12}$$

$$\mu_t(\boldsymbol{\alpha}_\chi) = \omega^{-1}\sum_{t'=t-\omega+1}^{t} \alpha_{\chi,t'}\,. \tag{13}$$

Notice that each measure is equivalent to the trace of the respective covariance matrix, i.e., the sum of the variances within each individual dimension.

One practical consideration to correctly compute these features on a rotation sequence $\boldsymbol{\alpha}_\rho$ concerns the fact that the space of quaternion is a double-covering group of SO(3). To avoid introducing fictitious variance, all quaternions are adequately inverted such that the scalar products of quaternions belonging to adjacent time steps are positive.

2) Linear Coefficient-Based Features: We construct a single-input multiple-output linear regression model of an observed trajectory $\boldsymbol{\alpha}_\chi$ within the pre-specified window as a function of a scalar time index t; i.e. $\tilde{\alpha}_{\chi,t} = \Theta_0 + t\,\Theta_1$.

We learn the model parameters Θ_0 and Θ_1 as a function of observed motion data $\hat{\boldsymbol{\alpha}}_\chi$. Parameter vector Θ_1 represents the Jacobian of the approximation function $\boldsymbol{J}_{\tilde{\boldsymbol{\alpha}}_\chi}$, and therefore contains the linear dependency of each output dimension w.r.t. the time variable. We thus use Θ_1 as our second feature type:

$$g_t(\boldsymbol{\alpha}) = (g_t(\boldsymbol{\alpha}_\pi)^\intercal, g_t(\boldsymbol{\alpha}_\rho)^\intercal)^\intercal\,,\text{ where} \tag{14}$$

$$g_t(\boldsymbol{\alpha}_\chi) = \Theta_1\left((\alpha_{\chi,t'})_{t'=t-\omega+1}^{t}\right)\,. \tag{15}$$

Intuitively speaking, feature values f_t quantify the total amount of movement happening *within* the window; whereas feature values g_t quantify the *net* amount of movement throughout the window.

One of the intended goals in this work is to define a model which is agnostic to the identities of the involved objects and, in general, invariant to properties other than the trajectories themselves. To eliminate the possibility that the model may learn different sets of parameters for the individual motions of objects A and B, we enforce that the same parameter values should be used. Because CRF models are log-linear, this is done by summing the respective features.

Finally, the state feature vector is constructed as

$$\phi_{\sigma,t}(x_t, \boldsymbol{y}) = \begin{pmatrix} \mathbb{I}[x_t = 1]\,(f_t(\boldsymbol{\alpha}) + f_t(\boldsymbol{\beta})) \\ \mathbb{I}[x_t = 1]\,(g_t(\boldsymbol{\alpha}) + g_t(\boldsymbol{\beta})) \\ \mathbb{I}[x_t = 1]\,f_t(\boldsymbol{\delta}) \\ \mathbb{I}[x_t = 1]\,g_t(\boldsymbol{\delta}) \end{pmatrix}. \quad (16)$$

The model thus counts a total of 22 parameters to learn (in addition to ω, which is not currently learned); 4 for the transitional feature vector and 18 for the state feature vector.

C. Model Usage

We use the model to perform two separate tasks:

1) Temporal Segmentation: The goal of this task is to detect hand-object interactions. Rather than tracking hand poses and using the CRF model directly, we determine hand-object interactions heuristically from the respective finger-object interactions. We do this because *a*) tracking fingers is easier than tracking hands and *b*) the finger-object relative motion is supposedly more stable during interaction, due to the contact point being on the finger.

In our formalization of the problem, each hand consists of 3 fingers (thumb, index and middle finger); we denote the interaction sequences for the hand and the fingers respectively as $\boldsymbol{h}$, $\boldsymbol{i}$, $\boldsymbol{j}$ and $\boldsymbol{k}$. $\boldsymbol{i}$, $\boldsymbol{j}$ and $\boldsymbol{k}$ are estimated as the most likely sequences of latent variables according to the CRF model when conditioned on the respective digit's tracked motion. $\boldsymbol{h}$ is heuristically computed using a majority voting scheme:

$$h_t = \mathbb{I}[i_t + j_t + k_t > 1]\,. \quad (17)$$

2) Constraint Extraction: The goal of this task is to extract a transferable abstract description of an assembly task as a sequence of constraints which should be created between pairs of objects. This is achieved using the described CRF model with one minor change: the temporal window used to compute the model features extends towards and until the end of the demonstration, rather than in the recent past.

To determine the order in which the objects are assembled, we apply the model to all pairs of objects in the scene and estimate the moment in time in which an object is assembled as the earliest moment in time where its interaction w.r.t. any other object is detected. The constraint between two objects is extracted as the average transformation between them during the detected interaction phases.

V. EVALUATION

A. Data Collection

We test our segmentation model on a number of demonstrations of complex sequential assembly tasks, as well as a few more basic cooperative object manipulation tasks:

- Toolbox — full assembly of a wooden toolbox.
- Toolbox_coop_[1–4] — cooperative partial assembly.
- Chair_[1–4] — full assembly of a plastic chair.

The toolbox is composed of 5 pieces, only 2 of which are involved in the cooperative partial assembly. The chair is composed of 6 pieces. Although precise object meshes were available to us for the toolbox, only approximate ones were obtainable for the chair.

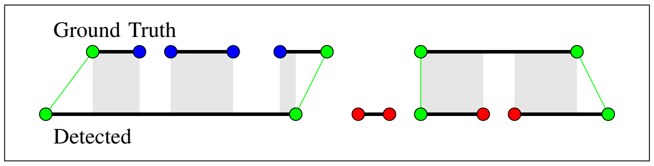

Fig. 2: Interaction phase scoring diagram. The green, red and blue dots represent 4 true positive, 4 false positive and 4 false negative transitions respectively. The transition match is based on the shaded *blocks*, which represent correctly detected interaction segments; for each block, the transitions of the corresponding interaction phases are matched if and only if the imaginary line segment which interpolates the transitions does not intersect any other block. True positives, false positives and false negatives are counted respectively as the number of matched transition pairs, the number of unmatched transitions in the detected sequence, and the number of unmatched transitions in the ground truth sequence.

Data collection is performed on 2 motion capture setups:

- Polhemus G4 magnetic tracking system — This setup tracks poses at 120Hz and its sensors are small enough to be attached to the individual fingers. This setup is used to produce all the demonstrations concerning the toolbox assembly.
- Optitrack vision based tracking system — This setup tracks poses at 120Hz. However, individual fingers are not tracked, and tracking is irregularly interrupted due to the possibility of occlusions[2]. This setup is used to produce all the demonstrations concerning the chair assembly.

B. Evaluation Criteria

Due to the necessity of tracking fingers for the temporal segmentation task, and to the role that precise object models have in the definition of a ground truth for the constraint extraction task (described below), quantitative evaluations are only available for the toolbox assembly demonstrations. The chair assembly demonstrations are used for qualitative benchmarking in scenarios where tracking suffers from occlusion.

We evaluate our model w.r.t. the two tasks as follows:

1) Temporal Segmentation: We define a scoring system which compares *transitions* of the detected interaction phases to the respective label phases. In this context, a transition is defined as the beginning or the end of an interaction phase. The scoring system counts the number of *true positive*, *false positive* and *false negative* transitions, and measures the temporal precision of true positive transition pairs as the standard deviation of their temporal difference (see Fig. 2).

2) Constraint Extraction: The constraint extraction task is evaluated by comparing the detected transformations to a corresponding ground truth transformation which is obtained

[2]Missing data is handled by the CRF model by removing the respective potentials or, equivalently, setting all the respective features as zero.

TABLE I: Scores for the temporal segmentation task. Tables (a)-(e) represent a demonstration each, and the rows and columns represent either the hands or the involved objects. Each cell contains a triplet consisting of the number of true positives, false positives and false negatives; and the standard deviation of the temporal difference [ms] between matching true positive transitions. Table (f) contains summary precision and recall statistics. The Toolbox demonstration is used to train the model when evaluating the Toolbox_coop_* demonstrations, and vice versa.

(a)

Toolbox_coop_1	Right Hand	Left Hand	Third Hand
Long Side 1	6 0 2 / 311	4 2 2 / 278	4 0 0 / 198
Short Side 1	2 0 0 / 744	4 0 4 / 361	4 0 0 / 217

(b)

Toolbox_coop_2	Right Hand	Left Hand	Third Hand
Long Side 1	4 0 2 / 710	2 2 0 / 859	4 0 0 / 416
Short Side 1	2 0 0 / 533	2 0 2 / 202	4 0 0 / 378

(c)

Toolbox_coop_3	Right Hand	Left Hand	Third Hand
Long Side 1	6 0 0 / 247	0 2 2 / —	4 0 0 / 159
Short Side 1	2 2 0 / 114	4 0 2 / 1458	2 0 2 / 436

(d)

Toolbox_coop_4	Right Hand	Left Hand	Third Hand
Long Side 1	6 0 0 / 307	2 2 0 / 13	4 0 0 / 218
Short Side 1	2 0 0 / 121	4 0 2 / 216	4 0 0 / 133

(e)

Toolbox	Long Side 1	Long Side 2	Short Side 1	Short Side 2	Handle
Right Hand	8 0 10 / 925	4 0 4 / 1359	10 0 12 / 876	10 0 8 / 843	6 0 10 / 1109
Left Hand	4 0 8 / 155	2 0 0 / 152	4 0 6 / 182	4 0 6 / 121	4 0 6 / 155

(f)

	Toolbox_coop_1	Toolbox_coop_2	Toolbox_coop_3	Toolbox_coop_4	Toolbox
Precision / Recall	92.31% / 70.59%	90.00% / 81.82%	81.82% / 75.00%	91.67% / 91.67%	100.00% / 44.44%

TABLE II: Scores for the constraint extraction task. Each table represents a demonstration, and the rows and columns represent the involved objects. Each cell contains the difference in translation [mm] and rotation [°] between the extracted and reference constraints for the corresponding object parts. The Toolbox demonstration is used to train the model when evaluating the Toolbox_coop_* demonstrations, and vice versa.

(a)

Toolbox_coop_1	Long Side 1	Short Side 1
Long Side 1	— / —	5.56 / 3.36
Short Side 1	8.90 / 3.36	— / —

(b)

Toolbox_coop_2	Long Side 1	Short Side 1
Long Side 1	— / —	8.63 / 2.54
Short Side 1	2.60 / 2.54	— / —

(c)

Toolbox_coop_3	Long Side 1	Short Side 1
Long Side 1	— / —	4.06 / 3.16
Short Side 1	7.40 / 3.16	— / —

(d)

Toolbox_coop_4	Long Side 1	Short Side 1
Long Side 1	— / —	5.74 / 2.80
Short Side 1	7.65 / 2.80	— / —

(e)

Toolbox	Long Side 1	Long Side 2	Short Side 1	Short Side 2	Handle
Long Side 1	— / —	7.49 / 1.99	1.52 / 1.88	5.85 / 2.16	7.56 / 0.68
Short Side 1	4.17 / 1.99	— / —	6.77 / 3.05	9.32 / 2.65	7.94 / 2.17
Short Side 2	2.84 / 1.88	6.76 / 3.05	— / —	9.55 / 1.62	6.09 / 1.48
Long Side 2	10.03 / 2.16	10.87 / 2.65	11.83 / 1.62	— / —	6.80 / 2.22
Handle	8.11 / 0.68	15.53 / 2.17	9.89 / 1.48	13.72 / 2.22	— / —

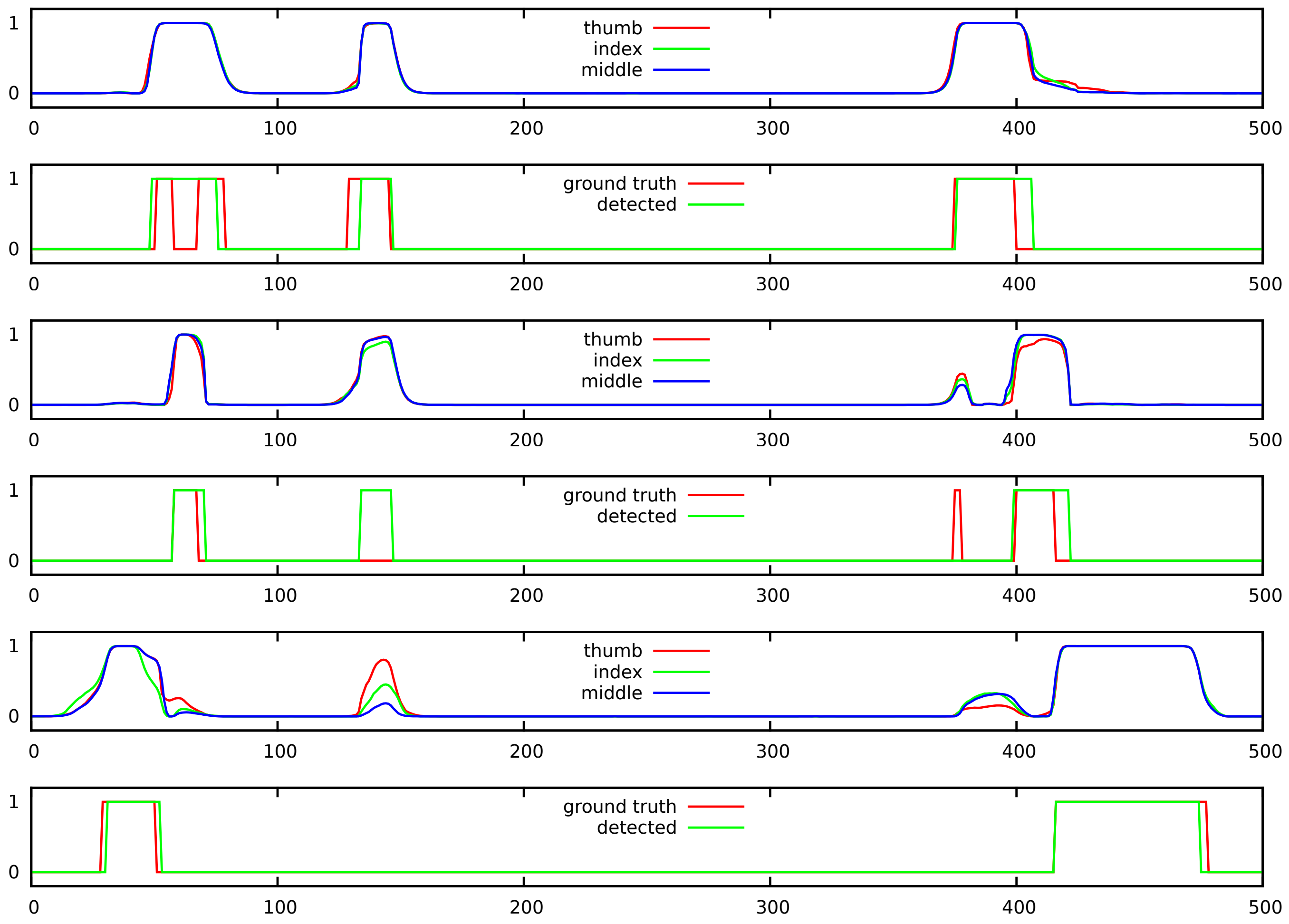

Fig. 3: Interaction phase detection profiles between all hands and the Long Side 1 object in the Toolbox_coop_1 demonstration. The first, second and third pairs of plots show respectively the interaction for the right, left and *third* (cooperative) hand. The first plot from each pair shows the marginal probabilities $p_t(x_t \,|\, \boldsymbol{y})$ for each finger-object pair; the second plot from each pair shows the detected hand-object interaction compared to the ground truth labeling. Upon inspection, one may notice, from the profiles, how the object is passed around between the hands.

by assembling the full piece of furniture in a simulated environment for high level robot programming [14]. For each pair of objects A and B (even those which are not connected *directly*), we compute the translation and angular difference between the detected transformation T_{AB} and its label $\tilde{T}_{AB}$ respectively as $\|(T_{AB}\tilde{T}_{BA})_\pi\|$ and $\angle[(T_{AB}\tilde{T}_{BA})_\rho]$. Notice that these measures are *not* necessarily symmetric w.r.t. the objects, thus the score of T_{AB} may differ from that of T_{BA}.

C. Discussion

Fig. 3 and Table I respectively show a selection of interaction profiles as detected by the model, and the previously defined scores for a set of toolbox assembly demonstrations. The low recall measure in the *Toolbox* scenario can be explained by a number of particularly brief undetected interactions which serve the purpose of displacing an object to make the assembly more comfortable. This type of manipulation, which does not achieve any sub-goal in the assembly task, does not appear as often in the partial assembly scenarios due to its simpler nature, and the lack of a necessity to re-organize the work-space between the execution of intermediate sub-tasks. Making the model more flexible in terms of the time-scales at which interactions are detected represents a sensible extension which would improve on this aspect.

Fig. 1 and Table II respectively show sample qualitative results on a toolbox and a chair assembly demonstration, and the previously defined scores for a set of toolbox assembly demonstrations. The performance on chair assembly tasks is qualitatively comparable to that on the toolbox ones, with the main exceptions being that *a*) the detection of a constraint may be delayed to a moment where sufficient motion of the involved object is observed, and *b*) the constraint generated at the end of a given demonstration is often left undetected. Both issues are supposedly due to a combination of missing data frames for a significant amount of time and the fact that less motion is generally available towards the end of a demonstration, to support the existence of a constraint.

Even when using tracking data suffering from occlusion, the constraint extracted from an object-object interaction phase is reasonable w.r.t. the assembly task at hand, meaning that the detection of false positive constraints (i.e. constraints with gross errors in position or orientation) does not generally occur.

VI. CONCLUSIONS

We have considered how to analyze complex sequential demonstrations of assembly tasks. In contrast with classical LfD approaches, whose main concern is often that of being able to replicate the performed motions, we focused our attention on detecting higher-level interactions between objects in the scene, which represent the real motivation behind each of the performer's motions.

The proposed discriminative model detects interactions based on the absolute and relative motion of the involved entities and is used to segment assembly demonstrations into the individual performed manipulation actions, as well as to generate the respective transferable assembly task description. The model shows promising quantitative and qualitative results w.r.t. the two proposed tasks, performing particularly well on data which features abundant motion.

The model is used to solve the constraint extraction task in [14], which presents an approach to high level robot programming based on an initial demonstration phase and a subsequent correction phase, where the user can apply manual modifications to the task representation which was learned by the system using a special purpose GUI (e.g. the user can provide constraints which were not autonomously detected, or fix imprecisions in the detected ones).

We identify a number of possible directions for future work: *a*) Incorporating contact-based features, which are themselves discriminative w.r.t. the physical interaction between entities, into the model would most likely improve general performance (at the additional cost of requiring object models and expensive geometric distance metric computations); *b*) the time-scale at which interaction phases are detected and deemed relevant is partially encoded within the window-length parameter ω, which is currently not learned from the data. The integration of features based on multiple window lengths is very likely to improve the detection of interactions which happen at different time-scales—such as the very short displacements which appear in our data when the human is *preparing* an object before assembly; *c*) the temporal segmentation can be further refined to reflect different modes within the same interaction phase, e.g. the usage of a tool, which can be detected as a single interaction phase, can be ulteriorly segmented into different motion modalities, such as "fetch", "use" and "lay_down"; and *d*) the manipulation segmentation can be leveraged to learn other high-level abstractions from demonstration, such a symbolic state description for each assembly sub-task and the corresponding symbolic transition models.

REFERENCES

[1] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4(4):343–355, 2012.

[2] B. Akgun, K. Subramanian, J. Shim, and A. L. Thomaz. Learning tasks and skills together from a human teacher. In *AAAI*, 2011.

[3] E. Aksoy, A. Abramov, F. Worgotter, and B. Dellen. Categorizing object-action relations from semantic scene graphs. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 398–405, May 2010.

[4] J. Almingol, L. Montesano, and M. Lopes. Learning multiple behaviors from unlabeled demonstrations in a latent controller space. In *Inter. Conf. on Machine Learning (ICML'13)*, Atlanta, USA, 2013.

[5] B. Argall, S. Chernova, and M. Veloso. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.

[6] J. Barbi, A. Safonova, J.-Y. Pan, C. Faloutsos, J. K. Hodgins, and N. S. Pollard. Segmenting motion capture data into distinct behaviors. In *Proceedings of the 2004 Graphics Interface Conference*, pages 185–194. Canadian Human-Computer Communications Society, 2004.

[7] S. Chiappa and J. R. Peters. Movement extraction by detecting dynamics switches and repetitions. In *Advances in neural information processing systems*, pages 388–396, 2010.

[8] A. Fod, M. J. Matari, and O. C. Jenkins. Automated derivation of primitives for movement classification. *Autonomous robots*, 12(1):39–54, 2002.

[9] O. Kroemer, C. Daniel, G. Neumann, H. van Hoof, and J. Peters. Towards learning hierarchical skills for multi-phase manipulation tasks. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2015.

[10] D. Kulić and Y. Nakamura. *From Motor to Interaction Learning in Robots*, chapter Incremental Learning of Full Body Motion Primitives. Springer, 2009.

[11] J. Lafferty, A. McCallum, and F. C. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[12] K. Lee, T.-K. Kim, and Y. Demiris. Learning action symbols for hierarchical grammar induction. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 3778–3782. IEEE, 2012.

[13] F. Meier, E. Theodorou, F. Stulp, and S. Schaal. Movement segmentation using a primitive library. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 3407–3412. IEEE, 2011.

[14] Y. Mollard, T. Munzer, A. Baisero, M. Toussaint, and M. Lopes. Robot programming from demonstration, feedback and transfer. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, 2015.

[15] S. Niekum, S. Chitta, A. G. Barto, B. Marthi, and S. Osentoski. Incremental semantically grounded learning from demonstration. In *Robotics: Science and Systems*, volume 9, 2013.

[16] S. Niekum, S. Osentoski, C. G. Atkeson, and A. G. Barto. Online bayesian changepoint detection for articulated motion models. *submitted to) IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[17] S. Niekum, S. Osentoski, G. Konidaris, and A. G. Barto. Learning and generalization of complex tasks from unstructured demonstrations. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5239–5246. IEEE, 2012.

[18] A. Pieropan, C. H. Ek, and H. Kjellstrom. Functional object descriptors for human activity modeling. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1282–1289. IEEE, 2013.

[19] S. Schaal. Is imitation learning the route to humanoid robots. *Trends in Cognitive Sciences*, 3(6):233–242, 1999.

[20] E. Vafeias and S. Ramamoorthy. Joint classification of actions and object state changes with a latent variable discriminative model. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 4856–4862, May 2014.

[21] B. Williams, M. Toussaint, and A. J. Storkey. Modelling motion primitives and their timing in biologically executed movements. In *Advances in neural information processing systems*, pages 1609–1616, 2008.

[22] A. S. Willsky, E. B. Sudderth, M. I. Jordan, and E. B. Fox. Sharing features among dynamical systems with beta processes. In *Advances in Neural Information Processing Systems*, pages 549–557, 2009.