Learning Complex Sequential Tasks from Demonstration: A Pizza Dough Rolling Case Study

Nadia Figueroa, Ana Lucia Pais Ureche and Aude Billard
Learning Algorithms and Systems Laboratory (LASA)
École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
Email: {nadia.figueroafernandez;lucia.pais;aude.billard}@epfl.ch

***Abstract*—This paper introduces a hierarchical framework that is capable of learning complex sequential tasks from human demonstrations through kinesthetic teaching, with minimal human intervention. Via an automatic task segmentation and action primitive discovery algorithm, we are able to learn both the high-level task decomposition (into action primitives), as well as low-level motion parameterizations for each action, in a fully integrated framework. In order to reach the desired task goal, we encode a task metric based on the evolution of the manipulated object during demonstration, and use it to sequence and parametrize each action primitive. We illustrate this framework with a pizza dough rolling task and show how the learned hierarchical knowledge is directly used for autonomous robot execution.**

I. Introduction

In order to teach a robot a complex task, involving a sequence of action primitives (atomic actions), such as pizza dough rolling, we must learn and discover a hierarchy of parameterizations for each level of the task. As opposed to typical LfD (Learning from Demonstration) approaches, where individual motions are learned independently and then the high-level composition of the task is either manually encoded or learned as a second step [1], we propose an integrated framework that is capable of extracting different levels of knowledge from the task, which are sufficient to parametrize the high-level task plan and low-level motion control of the robot, from complete demonstrations (See Figure 1).

Given successful demonstrations of the task by a skilled user via kinesthetic teaching, we begin by applying an automatic task segmentation and action primitive discovery algorithm to the raw data of the end-effector (Section II). This algorithm outputs a set of unique action primitives and their corresponding segmented data, as well as their transition probabilities. The latter is used to learn a probabilistic representation of the action sequencing involved in the task (Section III) and the former is used to extract individual action constraints to learn low-level motion parameterizations of each action (Section IV). Finally, the high-level action parameters (i.e. *rolling direction*) are defined by a task metric, which encodes the evolution of the manipulated object (Section V).

II. Autonomous Task Segmentation and Action Primitive Discovery

To *automatically segment and discover action primitives* we apply a method introduced in [2], which uses a Bayesian Non-Parametric approach for segmentation and clustering of time-series data. Bayesian nonparametrics allows for learning problems to be independent of *a priori* knowledge of model parameters such as the number of hidden states, cluster or

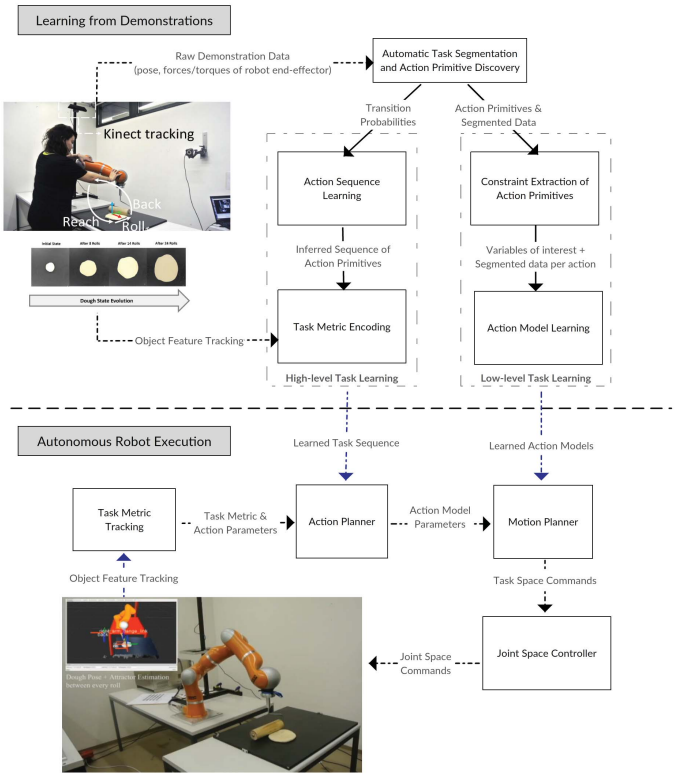

Figure 1: Learning Pipeline: Given a set of recorded demonstrations from a skilled user, we learn *high-level knowledge* such as the action primitive, their sequence and the task metric (left column) and *low-level knowledge* of the individual actions (right column). This knowledge is encoded as action/motion planners and used directly in robot execution.

mixtures. The algorithm used in this work is an extension of the Beta Process Hidden Markov Model (BP-HMM), with the capability of clustering similar actions subject to changes in scaling, translation and rotation, without any prior knowledge of the task, actions or the number of primitives involved. This is done by evaluating the similarity of the extracted Gaussian models from the HMM using a novel Spectral Polytope Covariance Matrix (SPCM) similarity function, which is invariant to any type of variation, such as translation, rotation and scale. This is highly useful in tasks such as dough rolling, since every rolling action is subject to a change in frame of reference *(rolling direction)* and scaling *(dough area)*.We apply the algorithm to the raw data-set $\xi = \{F, x\}$, where $F \in R^6$ is the wrench at the end effector and $x \in R^6$ is the end-effector position and orientation. In Figure 2, the extracted primitives and segmentations points are shown, we were able to extract 3 unique action primitives from this task: *reach*, *roll* and *reach back*, regardless of rolling direction, duration or force intensity.

978-1-4673-8370-7/16/$31.00 © 2016 IEEE

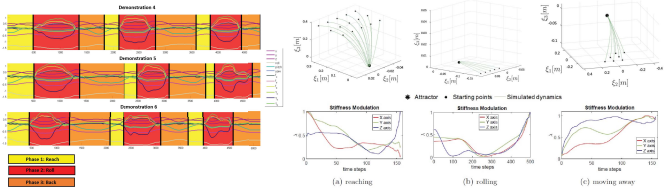

Figure 2: Extracted primitives and segmentation overlapped on each demonstration data ξ.

Figure 3: Learned motion dynamics for each phase

III. Action Sequence Learning

To avoid pre-defining or making assumptions on the primitives and the sequence, we use the transition probabilities π^j learned by the extended BP-HMM to find the correct sequence. For each j-th time series, a transition probability matrix is learned for the extracted primitives which are shared throughout all time series. The full set of N-time series represent multiple rolling demonstrations. In order to consider all possible π^j, we compute an average transition probability matrix: $\bar{\pi} = \frac{1}{\kappa N} \sum_{j=1}^{N} \pi^j$, where κ is the sticky parameter used to bias the HMM to match high self transitions and N is the total number of time-series used to extract the primitives. We can then construct a stationary markov chain of the primitives using $\bar{\pi}$ as the transition matrix. By computing the joint probabilities of each possible sequence p, we can infer the correct sequence by finding the $max(P(p_i)|i = 1, 2, 3) = \{1 \rightarrow 2 \rightarrow 3\}$, where i is the index of the unique action primitive.

IV. Action Constraints and Model Learning

Following the approach proposed in [3], we assume that all action primitives can be executed by the robot using a Cartesian impedance controller. Hence, we *extract* the *soft task constraints* from each action primitive, which correspond to the *variable of interest (F or x)* and *stiffness modulation*. Applied to the dough rolling task this method determined a position controller as the most suitable for the reaching actions and a hybrid force-position decomposition for the rolling action. We then learn a set of action models $s = \{1, 2, 3\}$ *corresponding to each action primitive*, $\psi_m^s = [x_m^s, F_m^s, K_m^s]$, which represent the parametrization of a Cartesian impedance controller, used to control a 7DOF robot arm, with the following control law: $\tau = J^T(K_m(x - x_m) + F_m)$. Where x_m, F_m, K_m are defined by their corresponding action models. $x_m \in R^6$ is the desired pose and is queried from an attractor-based coupled dynamical system (CDS) model, which encodes the position and orientation as two coupled autonomous dynamical systems (Figure 3). $F_m \in R^6$ is the desired force which is encoded as a function using a Gaussian Mixture Model (GMM) conditioned on the distance to the attractor. For dough rolling, we identified the vertical force applied on the dough as one of the tasks *soft constraints* (Figure 5). $K_m \in R^6$ corresponds to the stiffness and its encoded as a modulation function for a base stiffness value via a GMM (Figure 3).

V. Task Metric Encoding

The goal of this task is to achieve a desired size/shape of the pizza dough. From the demonstrations, we discovered that all rolling trajectories followed the same direction as the secondary principal component of the dough *(when fitting an ellipse to the shape)*. In addition, the ending points of the *reaching* segments were clustered at the beginning points of that direction and the ending of the *rolling* segments were clustered on the same direction but on the opposite side (see Figure 4). Thus, we use the secondary axis of a fitted ellipse on the dough as the main feature to parametrize the *rolling direction* and reached the desired circular shape.

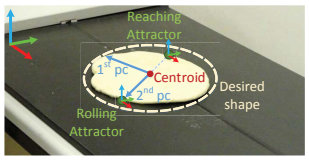

Figure 4: Positioning of the attractors wrt. dough shape

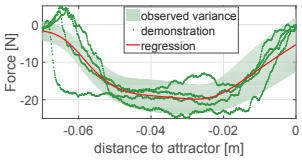

Figure 5: Model of the vertical force F_m^z

Control Mode	Metrics	Dough Type: Soft	Stiffer	Elastic
Hybrid	Def_{ratio}	2.6270 (0.0602)	2.585 (0.379)	2.5048 (0.1892)
	Iso_{quot}	0.9907 (0.0068)	0.9981 (0.0024)	0.9949 (0.0074)
	Num_{rolls}	12.3333 (1.5275)	38.66 (3.055)	76.3333 (3.5119)
	Max_{force}	38.3540 (3.1945)N	41.126 (1.899)N	45.5943 (3.4927)
Pos-Stiff	Def_{ratio}	2.5240 (0.2507)	2.5559 (0.2128)	2.8250 (0.3952)
	Iso_{quot}	0.9940 (0.0047)	0.9855 (0.0110)	0.9932 (0.0072)
	Num_{rolls}	14.6667 (1.5275)	46.6667 (4.0415)	118.6667 (19.1398)
	Max_{force}	71.0843 (1.7159)N	78.9383 (2.8563)N	83.0120 (3.2562)N

Table I: Evaluation of control modes. *(Hybrid)* is our proposed framework, *(Pos-Stiff)* is a standard position controller with fixed high stiffness values. The values in the metrics are the **mean (std.)** of 3 trials per control mode/dough consistency.

VI. Framework Evaluation

We evaluate our framework with 2 different control modes: *(hybrid)* being our approach where we control for position and modulate force/stiffness during interaction and *(pos-stiff)* is a standard position controller that will follow the desired trajectories with high stiffness, but with a hand-tuning in the position for the rolling phase (in order to achieve flattening of the dough). We ran each control mode 3 times for 3 different dough consistencies: (i) very *soft* (freshly made) dough, (ii) a bit *stiffer* dough (once it cooled down) and (iii) a hard dough, almost *elastic*. The task performance is then evaluated with the following measures:

1) Def_{ratio}: Deformation achieved, computed by dividing the ellipse area ratio between last/first roll.
2) Iso_{quot}: The isoperimetric quotient evaluates the obtained roundness of the resulting deformed dough.
3) NR: # of rolls needed to achieve the target area ($0.025m^2$).
4) Max_{force}: Maximum force applied to the dough.

Using the following task settings across trials: initial dough area $\approx 0.01m^2$, dough weight $\approx 0.5kg$ and goal area $\approx 0.025m^2$ we gathered the results presented in Table I, with *(hybrid)* achieving more consistent shapes (Iso_{quot}) than *(pos-stiff)*. Both control modes achieve the desired size. However, since *(pos-stiff)* uses a fixed high stiffness and hand-tuned position, it reached extremely high values during contact (70-80 N), which causes the robot to become unstable. Whereas *(hybrid)* reached the target size/shape with less rolls and safe force levels (40N), as the ones learned from human demonstrations.

VII. Future Work

Our ongoing work is focused on autonomously learning the relevant features of the manipulated object that evolve as en effect of the action primitives. We will then apply this framework to different tasks where the manipulated object is subject to some sort of deformation be it shape, size, color, such as painting, batter mixing or egg beating.

Acknowledgements

This work was supported by the EU FP7 Project *RoboHow* (Grant Agreement Number 288533).

References

[1] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469 – 483, 2009.

[2] N. Figueroa and A. Billard, "On discovering structure in heterogeneous, unstructured and sequential tasks from demonstrations," in *In preparation*, 2015.

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