Approaches for Learning Human-like Motor Skills which Require Variable Stiffness During Execution

Petar Kormushev, Sylvain Calinon and Darwin G. Caldwell

***Abstract*— Humans employ varying stiffness in everyday life for almost all human motor skills, using both passive and active compliance. Robots have only recently acquired variable passive stiffness actuators and they are not yet mature. Active compliance controllers have existed for a longer time, but the problem of automatic determination of the necessary compliance to achieve a task has not been thoroughly studied. Teaching humanoid robots to apply variable stiffness to the skills they acquire is vital in order to achieve human-like naturalness of the execution. Also, using adaptive compliance can help to increase the energy efficiency. This paper compares two different approaches that allow robots to learn human-like skills which require varying stiffness during execution. The advantages and disadvantages of each approach is discussed and demonstrated with various experiments on an actively-compliant Barrett WAM robot.**

keywords: variable stiffness, motor skills, imitation learning, reinforcement learning, programming by demonstration, kinesthetic teaching, pHRI

I. INTRODUCTION

Endowing robots with human-like abilities to perform motor skills in a smooth and natural way will greatly promote the use of robots in everyday life. One particularly intersting aspect of implementing natural, human-like motor skills for robots is using variable stiffness during the skill execution.

Humans employ varying stiffness in almost all human motor skills, using both passive and active compliance. Robots have only recently acquired variable passive stiffness actuators and they are not yet mature. Active compliance controllers have existed for a longer time, but the problem of automatic determination of the necessary compliance to achieve a task has not been thoroughly studied. Teaching humanoid robots to apply variable stiffness to the skills they acquire is vital in order to achieve human-like naturalness of the execution. Also, using adaptive compliance can help to increase the energy efficiency.

In this paper we study the question how a robot can learn variable-stiffness skills *without* manually being programmed or taught explicitly.

Many approaches exist for teaching robots motor skills, but only few of them have been demonstrated to be able to learn variable stiffness skills.

When direct physical contact is possible, *kinesthetic teaching* offers a user-friendly and intuitive method to demonstrate new skills to a robot by manually guiding the robot's arms through the motion [1]. Different than standard teleoperation techniques, it allows the user to feel firsthand the limitations of the robot's body (e.g. small number of DOF, joint limits, short limbs, singularities, too big stiffness, too slow movement, self-collision, etc.), which is especially important on highly-dynamic tasks [2].

Lightweight robots offer new perspectives to augment the personal capabilities of the robot through kinesthetic teaching, where control at a torque level can be used to let the user move the robot as if it had no weight, and as if no motors were in its articulations. Through a gravity compensation actuation, the user can concentrate on the task to demonstrate by grasping (and re-grasping) the robot in ways that are natural and efficient for him/her to execute the task. Successful applications of this torque-based kinesthetic teaching process have been recently developed for the Barrett WAM and KUKA/DLR robotic arms [2], [3].

Variable Stiffness Actuators (VSA) have been created in the last few years which pose new challenges for the learning algorithms. Also, the control algorithms now have to deal with both active and passive compliance.

In this paper we examine two approaches for learning variable stiffness skills: the *Extracted variability* approach, and the *Self-learned variability* approach.

The *Extracted variability* approach belongs to the Programming-by-Demonstration [1] [4] (a.k.a. Imitation Learning) domain. It is based on extracting the variance among multiple task demonstrations and using this variance to infer the required variability of the stiffness during task reproduction.

The *Self-learned variability* approach belongs to the Reinforcement Learning [5] (a.k.a. self-learning) domain. In this approach the robot explores by itself variations of the stiffness during multiple trials and discovers what variability is required for successful task reproduction.

First we describe each of the two approaches, then we illustrate them with experiments, and finally we discuss their advantages and disadvantages.

II. APPROACHES FOR LEARNING VARIABLE STIFFNESS

In this section we overview two approaches for learning variable stiffness skills: the *Extracted variability* approach, and the *Self-learned variability* approach.

A. Extracted variability approach

In this approach, we refer to the kinematic redundancy of the robot when the robot possesses an infinite number of generalized inverse control strategies, see e.g. [6], [7].

The authors are with the Advanced Robotics Department, Italian Institute of Technology (IIT), 16163 Genova, Italy. {petar.kormushev,sylvain.calinon, darwin.caldwell}@iit.it.

We refer to task redundancy when the task can be achieved through an infinite number of solutions, see e.g. [8]. We take the perspective that both the robot and task redundancies can be exploited to regulate the dynamics of the movement and the stiffness of the robot during reproduction.

After having observed several demonstrations of a similar task, the robot creates a compact model of the skill, by taking into account the variations and correlations observed along the movement. If a part of the movement was consistent across the different trials, this part of the task should probably be reproduced in this specific manner. On the other hand, if a large variability was observed among the different demonstrations, reproducing a specific reference trajectory may be irrelevant to fulfill the task requirements.

During reproduction, the robot is using this information to set an adequate stiffness that will fulfill the task constraints, which allows to simultaneously consider other constraints. There are several situations where the interaction can benefit from the variability and correlations of the task: (i) to let the user physically move the robot while reproducing the task; (ii) to let the robot modify the generalized trajectory to adopt gestures that are safer for a user who is close to the robot.

The positional constraints of the demonstrated skill are represented as a mixture of dynamical systems encoding robustly position trajectories. The *Dynamic Movement Primitives* (DMP) framework originally proposed by Ijspeert *et al* [9], and further extended in [10], [11] (see [12] for a discussion on the similarities of the proposed controller with DMP) is extended by considering a full matrix $K_i^{\mathcal{P}}$ associated with each of the K primitives (or states) instead of a fixed $\kappa^{\mathcal{P}}$ gain. This allows us to take into consideration variability and correlation information along the movement for learning and reproduction.

To encode the positional profile of the task, an extended DMP framework is used to take into consideration the variability of the movement during the learning process.

M examples of a skill are demonstrated to the robot in slightly different situations. Each demonstration $m \in \{1,\dots,M\}$ consists of a set of T_m positions x, velocities $\dot{x}$ and accelerations $\ddot{x}$ of the end-effector in Cartesian space, where each position x has $D = 3$ dimensions. A dataset is formed by concatenating the $N = \sum_{m=1}^{M} T_m$ datapoints $\{\{x_j, \dot{x}_j, \ddot{x}_j\}_{j=1}^{T_m}\}_{m=1}^{M}$. A desired acceleration is computed based on a mixture of K proportional-derivative systems

$$\hat{\ddot{x}} = \sum_{i=1}^{K} h_i(t)\Big[K_i^{\mathcal{P}}(\mu_i^X - x) - \kappa^{\mathcal{V}}\dot{x}\Big]. \qquad (1)$$

Parts of the movement where the variations across the different demonstrations are large indicate that the reference trajectory does not need to be tracked precisely. By using this information, the controller can focus on the other constraints of the task such as collision avoidance. On the other hand, parts of the movement exhibiting strong invariance across the demonstrations should be tracked precisely, i.e., the stiffness used to track the position errors needs to be high.

The superposition of basis vector fields is determined in (1) by an implicit time dependency, but other approaches

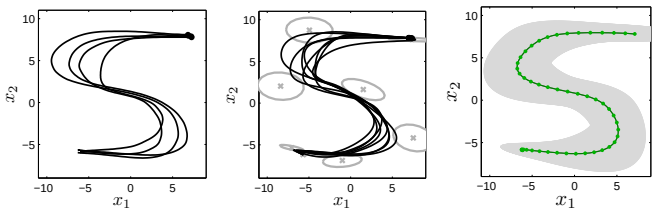

Fig. 1. Illustration of the learning and retrieval processes. *Left:* Four examples of the task provided as demonstrations. *Center:* Learned model represented by Gaussians $\mathcal{N}(\mu_i^X, \Sigma_i^X)$, and multiple reproduction attempts. *Right:* The trajectory in green lines shows a generalized reproduction attempt. The points show positions at constant time intervals.

using spatial and/or sequential information could also be used [13], [14]. Similarly to DMP, a decay term defined by a canonical system $\dot{s} = -\alpha s$ is used to create an implicit time dependency $t = -\frac{ln(s)}{\alpha}$, where s is initialized with $s = 1$ and converges to zero. We define a set of Gaussians $\mathcal{N}(\mu_i^T, \Sigma_i^T)$ in time space τ, with centers μ_i^T equally distributed in time, and variance parameters Σ_i^T set to a constant value inversely proportional to the number of states. α is initially fixed depending on the duration of the demonstrations. The weights $h_i(t)$ are defined by

$$h_i(t) = \frac{\mathcal{N}(t;\ \mu_i^T, \Sigma_i^T)}{\sum_{k=1}^{K}\mathcal{N}(t;\ \mu_k^T, \Sigma_k^T)}. \qquad (2)$$

By determining the weights through the decay term s, the system will sequentially converge to the set of attractors in Cartesian space defined by μ_i^X. The centers μ_i^X in task space and stiffness matrices $K_i^{\mathcal{P}}$ are learned from the observed data, either incrementally or in a batch mode (through least-squares regression). For example, parts of the movement where the variations between the demonstrations are high indicate that the reference trajectory does not need to be tracked precisely. By using this information, the controller can focus on the other constraints of the task such as moving away from the user. On the other hand, parts of the movement exhibiting strong invariance among the demonstrations should be tracked precisely, i.e., the stiffness used to track the position errors needs in this case to be high.

In a batch mode, by concatenating the training examples in a matrix $Y = [\ddot{x}\frac{1}{\kappa^{\mathcal{P}}} + \dot{x}\frac{\kappa^{\mathcal{V}}}{\kappa^{\mathcal{P}}} + x] \in \mathbb{R}^{N\times D}$, and by concatenating the corresponding weights computed with (2) in a matrix $H \in \mathbb{R}^{N\times K}$, we can write the linear equation $Y = H\mu^X$, with $\mu^X \in \mathbb{R}^{K\times D}$ representing the concatenated attractor centers μ_i^X. The least-squares solution to estimate the attractor centers is then given by $\mu^X = H^{\dagger}Y$, where $H^{\dagger} = (H^{\top}H)^{-1}H^{\top}$ is the pseudoinverse of H. By defining a desired range of stiffness values $[\kappa_{\min}^{\mathcal{P}}, \kappa_{\max}^{\mathcal{P}}]$, we define the stiffness and damping gains for the estimation of the parameters as $\kappa^{\mathcal{P}} = \kappa_{\min}^{\mathcal{P}} + \frac{\kappa_{\max}^{\mathcal{P}} - \kappa_{\min}^{\mathcal{P}}}{2}$ and $\kappa^{\mathcal{V}} = 2\sqrt{\kappa^{\mathcal{P}}}$.

To take into account variability and correlation along the movement and among the different demonstrations, we compute for each state $i \in \{1,\dots,K\}$ the residual errors of the least-squares estimation, in the form of covariance

matrices

$$\Sigma_i^x = \frac{1}{N}\sum_{j=1}^{N}(Y'_{j,i}-\bar{Y}'_i)(Y'_{j,i}-\bar{Y}'_i)^\top \quad \forall i \in \{1,\ldots,K\} \text{ where} \tag{3}$$

$$Y'_{j,i} = H_{j,i}(Y_j - \mu_i^x). \tag{4}$$

In the above equation, $\bar{Y}'_i$ is the mean of Y'_i over the N datapoints. $\mathcal{N}(\mu_i^x, \Sigma_i^x)$ thus describes a Gaussian in Cartesian space x. The set of K Gaussians defines the sequence of virtual attractor points in Cartesian space that the system will try to reach, where each attractor encapsulates variability and correlation information. The residuals terms of the regression process are then used to estimate the stiffness matrices $K_i^{\mathcal{P}}$ in Eq. (1) through eigencomponents decomposition $K_i^{\mathcal{P}} = V_i D_i V_i^{-1}$, with

$$D_i = \kappa^{\mathcal{P}}_{\min} + (\kappa^{\mathcal{P}}_{\max} - \kappa^{\mathcal{P}}_{\min})\frac{\lambda_i - \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}}. \tag{5}$$

In the above equation, λ_i and V_i are the concatenated eigenvalues and eigenvectors of the inverse covariance matrix $(\Sigma_i^x)^{-1}$. The basic idea is to determine a stiffness matrix proportional to the inverse of the observed covariance. For example, if high variability is observed, stiffness will become low as the tracking does not need to be precise. If D_i in (5) is set to λ_i, the eigencomponents decomposition gives $K_i^{\mathcal{P}} = (\Sigma_i^x)^{-1}$. We rescale D_i to obtain stiffnesses in the desired range $[\kappa^{\mathcal{P}}_{\min}, \kappa^{\mathcal{P}}_{\max}]$ (determined by the user and hardware's limitation) based on the initial range of eigenvalues $[\lambda_{\min}, \lambda_{\max}]$ (determined by the variability within the motion and among several demonstrations). The minimal stiffness limit allows us to set a degree of compliance that still allows the system to move (e.g. to limit friction effects), while the maximum stiffness limit can be fixed depending on the robot capabilities, or based on desired safety limits.

Fig. 1 illustrates the approach with a 2-dimensional example. In the top-right graph, we see that the trajectories reproduced stochastically from the learned model show different levels of variability along the trajectory. This variability shows similar characteristics to the one of the training set. We also see that the reproduced trajectories share similar smoothness to the demonstrated trajectories. The trajectories have been represented with dots showing the position of the robot at fixed time intervals.

B. Self-learned variability approach

Sometimes it is impossible for a non-expert human teacher to demonstrate a task with variable stiffness. For example, baseball batting - it would be very difficult for a non-trained person to demonstrate to the robot a good way to perform the task. Even more, with the requirement of multiple such demonstrations, it becomes practically impossible to use the Extracted variability approach.

For such situations, a self-learning approach might be more appropriate. In this method, first the positional profile of the task is (only roughly) demonstrated. After that, the robot uses exploration algorithm to find an appropriate stiffness level for every part along the demonstrated trajectory.

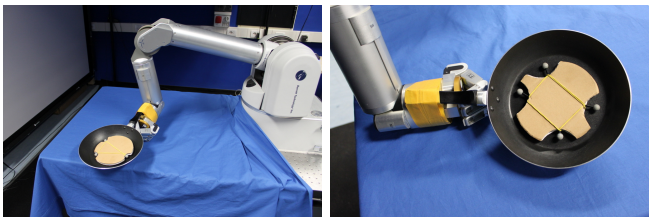

Fig. 2. Experimental setup for the *Pancake-Flipping* task. A torque-controlled 7-DOF Barrett WAM robot learns to flip pancakes in the air and catch them with a real frying pan attached to its end-effector. Artificial pancakes with passive reflective markers are used to evaluate the performance of the learned policy.

Since such dynamic motor skills are very task-specific. In this study we show the results from a pancake flipping task.

The self-learning approach represents a movement as a superposition of basis force fields, where the model is initialized from imitation. Reinforcement Learning (RL) is then used to adapt and improve the encoded skill by learning optimal values for the policy parameters. The policy parameterization allows the RL algorithm to learn the coupling across the different motor control variables and, in effect, variable (active) compliance.

To learn new values for the full stiffness matrices, we use the state-of-the-art EM-based RL algorithm called PoWER developed by Kober and Peters [15]. PoWER inherits from EM algorithm two major advantages over policy-gradient-based approaches: firstly, PoWER does not need a learning rate, unlike policy-gradient methods; secondly, PoWER can be combined with importance sampling to make better use of the previous experience of the agent in the estimation of new exploratory parameters.

Similar to policy gradient RL, PoWER uses a parameterized policy and tries to find values for the parameters which maximize the expected return of rollouts (also called episodes or trials) under the corresponding policy. In our approach the policy parameters are represented by the elements of the full stiffness matrices $K_i^{\mathcal{P}}$ and the attractor vectors μ_i^x.

The return of a rollout τ is given by the undiscounted cumulative reward $R(\tau) = \sum_{t=1}^{T} r(t)$, where T is the duration of the rollout, and $r(t)$ is the reward received at time t, defined differently according to the goal of the task.

In general, as an instance of an EM algorithm, PoWER estimates the policy parameters θ such as to maximize the lower bound on the expected return from following the policy. The policy parameters θ_n at the current iteration n are updated to produce the new parameters θ_{n+1} using the following rule (see also [16])

$$\theta_{n+1} = \theta_n + \frac{\left\langle (\theta_k - \theta_n) R(\tau_k) \right\rangle_{w(\tau_k)}}{\left\langle R(\tau_k) \right\rangle_{w(\tau_k)}}. \tag{6}$$

In the above equation, $(\theta_k - \theta_n) = \Delta\theta_{k,n}$ is a vector difference which gives the relative exploration between the

policy parameters used in the k-th rollout and the current ones. Each relative exploration $\Delta\theta_{k,n}$ is weighted by the corresponding return $R(\tau_k)$ of rollout τ_k, and the result is normalized using the sum of the same returns.

In order to minimize the number of rollouts which are needed to estimate new policy parameters, we use a form of importance sampling technique adapted for RL [5], [15] and denoted by $\langle\cdot\rangle_{w(\tau_k)}$ in Eq. (6). It allows the RL algorithm to re-use previous rollouts τ_k and their corresponding policy parameters θ_k during the estimation of the new policy parameters θ_{n+1}. The importance sampler we use is defined as

$$\Big\langle f(\theta_k, \tau_k) \Big\rangle_{w(\tau_k)} = \sum_{k=1}^{\sigma} f(\theta_{\mathrm{ind}(k)}, \tau_{\mathrm{ind}(k)}), \qquad (7)$$

where σ is a fixed parameter denoting the number of rollouts used by the importance sampler, and $\mathrm{ind}(k)$ is an index function which returns the index of the k-th best rollout in the list of all past rollouts sorted by their corresponding returns, i.e. for $k = 1$ we have $\mathrm{ind}(1) = \mathrm{argmax}_i\, R(\tau_i)$, and $R(\tau_{\mathrm{ind}(1)}) \geq R(\tau_{\mathrm{ind}(2)}) \geq ... \geq R(\tau_{\mathrm{ind}(\sigma)})$. The effect of the importance sampler is significant because it allows the RL algorithm to re-use the top-σ best rollouts so far in order to calculate the new policy parameters. This helps to reduce the number of required rollouts and makes the algorithm applicable to online learning, which we demonstrate with the *Pancake-Flipping* task described in Section III-D.

III. EXPERIMENTS

A. Experimental setup

The proposed methods are evaluated on two human-robot interaction experiments, which aim to teach the robot to perform two tasks: *ironing task* and *door opening task*. All experiments are conducted using a torque-controlled 7-DOF *Barrett WAM* robotic arm with 3-finger Barrett Hand attached.

The demonstrations needed for learning the positional profile of each task are recorded via kinesthetic teaching, i.e. a human demonstrator is holding the arm of the robot and manually guiding the robot to execute the task. During this, the WAM robot is put in a gravity-compensation mode which allows the user to move it effortlessly.

At this point, an autonomous reproduction of the task can be attempted by the robot using the extracted positional constraints.

The demonstration phase of the teaching process is shown in Fig. 3 for the ironing task and the door opening task. Each task is encoded using the proposed model, by fixing the number of states (or primitives) empirically with respect to the length of the demonstrations. For the ironing task, 6 demonstrations were provided for the positional profile, and for the door opening task - 5 demonstrations.

B. Ironing task

The ironing task poses a wide variety of challenges for robotics. These challenges range from the detection, grasping and folding of clothes, to position and orientation analysis

Fig. 3. Teaching the positional profiles of the ironing task and door opening task.

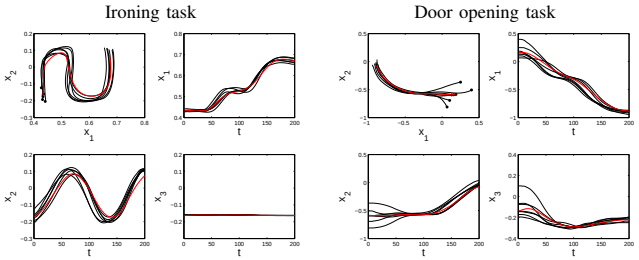

Fig. 4. Positional profiles of the ironing task and door opening task. Demonstrations (in black) and separate reproductions (in red) based on the learned individual profiles. The position axis x_1 and x_2 are in the horizontal plane, and x_3 is the vertical axis.

of the iron depending on the regions of garment and on the proximity to an operator. Here we concentrate on the simultaneous consideration of position information for the ironing skill. For example, while the iron is being moved, it should be pushed down towards the table top, to press the clothes well. In addition, the exerted force on the iron should be stronger when the tip of the iron is moving forward, and weaker when the iron is moving backward, to avoid creating wrinkles on the clothes.

1) Learning the task profiles: The ironing skill imposes varying positional constraints throughout the execution of the task. For example, the path that the iron should follow is more constrained in the vertical axis than in the horizontal plane because it is more important to have the iron in contact with the table than to follow a specific path on the table. These varying positional constraints are reflected by the collected data from the demonstrations and incorporated in the task profile.

Fig. 4 shows the obtained positional and profiles of the ironing task after the teaching process. Fig. 5 demonstrates the ability of the proposed method to encode the variable positional task constraints and to use adaptive stiffness gain matrix to perform adequate task reproductions. In particular, the learned model shows that it is more important to track the movement in the vertical direction than in the other two directions of the horizontal plane (nearly flat ellipsoids in the second graph). As a result, the stiffness matrices have an elongated shape reflecting this property.

The two simulated perturbation experiments demonstrate that the robot learned to keep its end-effector stiff in vertical direction and compliant (soft) in horizontal direction. As a result, when applying a constant force to the end-effector during reproduction, the deformation is stronger if the force is parallel to the table than if the force is vertical. The

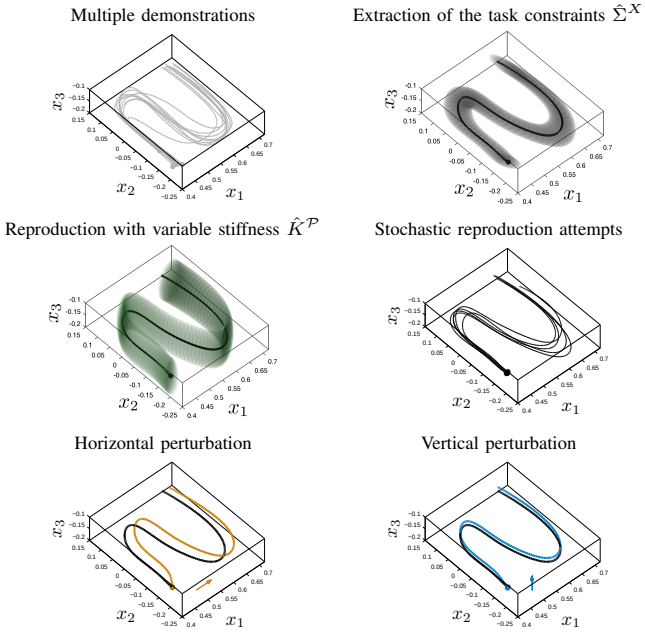

Fig. 5. Learned positional profile for the ironing task. *Top:* Extracting task constraints from demonstrations through the residuals of the regression process. Adaptive stiffness gain matrix computed from the residuals information. The covariance matrices $\hat{\Sigma}^X = \sum_{i=1}^{K} h_i(t)\Sigma_i^X$ and stiffness gain matrices $\hat{K}^{\mathcal{P}} = \sum_{i=1}^{K} h_i(t)K_i^{\mathcal{P}}$ are represented with grey and green ellipsoids respectively. *Bottom:* Stochastic reproductions of the movement. Two perturbed reproduction attempts: one with applying a constant force parallel to the table at the end-effector of the robot, and another with applying a constant force with the same amplitude, but perpendicular to the table.

covariance matrices $\hat{\Sigma}^X$ and stiffness gain matrices $K^{\mathcal{P}}$ are respectively depicted with grey and green ellipsoids in Fig. 5. The variability of the demonstrations reflects the important characteristics of the task. The model correctly encapsulates these variations through the set of covariance matrices $\hat{\Sigma}^X$ estimated through the residuals of the regression process. A variable stiffness gain matrix $\hat{K}^{\mathcal{P}}$ is automatically calculated in order to fulfill the learned task constraints.

C. Door opening task

The second task consists of opening a door which has a horizontal bar that needs to be pushed with a larger force than a standard door handle to open it, as shown in Fig. 2. In this task, the robot first has to push with its hand the horizontal bar downwards (which unlocks the door) and then to push forward and to the right in an arc-shaped trajectory to open the door wider.

1) Learning the task profiles: The teaching process for the door opening task is shown in Fig. 3. The individual positional and force profiles obtained from the demonstrations are shown in Fig. 4.

Regarding the positional profile of the task, the position of the hand needs to be more constrained when it pushes on the handle than during its approach, in order to bring it to the desired position of the horizontal bar. These varying positional constraints are reflected by the collected data and learned by the proposed method.

D. Pancake-Flipping task

The real-world evaluation of the Self-learning variability approach is done on a dynamic *Pancake-Flipping* task. The goal of the task is to toss a pancake in the air so that it rotates 180 degrees before being caught. Due to the complex dynamics of the task, it is unfeasible to try to learn it directly using *tabula rasa* RL. Instead, a person presents a demonstration of the task first, which is then used to initialize the policy.

1) Experimental setup: The experimental setup is shown in Fig. 2. The experiment is conducted with a torque-controlled *Barrett WAM* 7 DOFs robotic arm. Using a gravity-compensation controller, the *Pancake-Flipping* task is first demonstrated via kinesthetic teaching. The number of states is fixed at 8, which is determined empirically by examining the quality of the initial reproduced trajectories with different number of states.

Custom-made artificial pancakes are used, which have 4 highly-reflective passive markers, in order to track both the position and the orientation of the pancakes during the task execution (See Fig. 2). For easier visual inspection, the two sides of the pancakes are colored in different colors - white and yellow. The pancake weights only 26 grams, which makes it susceptible to air flow influence and makes its motion less predictable.

The pancake's position and orientation are tracked by a marker-based *NaturalPoint OptiTrack* motion capture system with 12 cameras. It tracks the position x^p and orientation (q^p in quaternion representation, M^p in direction cosine matrix representation) of the pancake at a rate of 30 frames per second.

The return of a rollout τ is calculated from the timestep reward $r(t)$. It is defined as a weighted sum of two criteria (orientational reward and positional reward), which encourage successful flipping and successful catching of the pancake

$$r(t_f) = w_1\left[\frac{\arccos(v_0.v_{t_f})}{\pi}\right] + w_2 e^{-||x^p - x^F||} + w_3 x_3^M, \quad (8)$$

where t_f is the moment when the pancake passes with downward direction the horizontal level at a fixed height Δ_h above the frying pan's current vertical position, v_0 is the initial orientation vector of the pancake (unit vector perpendicular to the pancake), v_{t_f} is the orientation vector of the pancake at time t_f, x^P is the position of the pancake at time t_f, x^F is the position of the frying pan at time t_f, and x_3^M is the maximum reached altitude of the pancake. The first term is maximized when the pancake's orientation vector at time t_f goes in the opposite direction to the initial orientation vector, which corresponds to a successful flip. The second term is maximized when the pancake lands in the center of the frying pan. The weights we use are $w_1 = w_2 = w_3 = 0.5$. For all other time steps $t \neq t_f$ we define $r(t) = 0$.

The learning process is based on the PoWER algorithm implementation provided by Kober *et al* [16]. $\sigma = 6$ is used as parameter for the importance sampler. The parameters θ_n

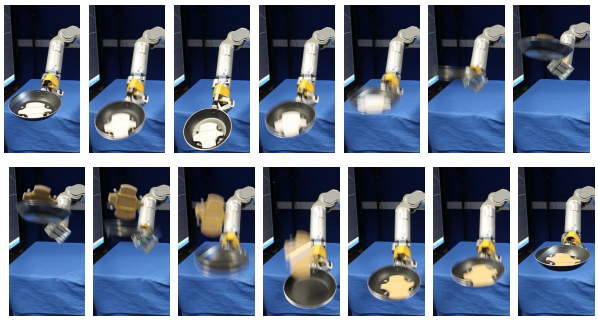

Fig. 6. Sequence of video frames showing a successful pancake flipping (after 50 rollouts), performed on the WAM robot.

for the RL algorithm are composed of two sets of variables: the first set contains the full 3×3 stiffness matrices $K_i^{\mathcal{P}}$ with the positional error gains in the main diagonal and the stiffness gains in the off-diagonal elements; the second set contains the vectors μ_i^X with the attractor positions for the primitives. The RL algorithm is stopped when a successful and reproducible pancake flipping is achieved with return $R(\tau) \geq 0.9$.

IV. DISCUSSION

The three presented experiments demonstrate the ability of the proposed methods to successfully encode and reproduce the variable stiffness needed to perform the corresponding motor skills successfully. For example, the ironing task requires variable stiffness in vertical and horizontal direction during the reproduction. Also, it requires exerting a variable force with constant direction on an object which does not move, regardless of the exerted force (the table top in this case). The door opening task, on the other hand, focuses on exerting a large force at a particular part of the trajectory, and also varying the direction of the force during the task reproduction. In both cases, the tasks are difficult to realize using fixed stiffness, which shows the advantage of incorporating variable stiffness information in the task model.

This paper does not deal with the concrete implementation of stiffness/compliance: the discussed points apply to both active and passive variable stiffness implementations.

The robot remains gravity-compensated throughout all the interactions presented in the paper for both demonstration and reproduction phases. During reproduction, this allows the user to physically move the robot using the redundancy of the robot's structure and the redundancy of the task. The same property can be exploited to improve the safety of the human-robot interaction by using the available redundancy to avoid hitting obstacles or other people close to the robot.

V. CONCLUSION

The presented methods allow a robot to learn motor skills requiring variable stiffness during execution.

In the Extracted variability approach, the user teaches the robot how to perform a task by providing multiple demonstrations of the skill he/she wishes to transfer. After extraction of the task constraints, the robot reproduces the task by automatically selecting an adequate level of compliance to reproduce the essential positional characteristics of the skill. We demonstrated the feasibility of the approach with an ironing task and a door opening task.

In the Self-learning variability approach, RL is used to find appropriate stiffness. The pancake flipping task is a good example for this approach, because of its dynamism and wide range of stiffness required.

REFERENCES

[1] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Secaucus, NJ, USA: Springer, 2008, pp. 1371–1394.

[2] P. Kormushev, S. Calinon, and D. G. Caldwell, "Robot motor skill coordination with EM-based reinforcement learning," in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, October 2010.

[3] A. Albu-Schaeffer, S. Haddadin, C. Ott, A. Stemmer, T. Wimboeck, and G. Hirzinger, "The DLR lightweight robot - Design and control concepts in human environments," *Industrial Robot*, vol. 34, no. 5, pp. 376–385, 2007.

[4] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, 2009.

[5] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, ser. Adaptive computation and machine learning. Cambridge, MA, USA: MIT Press, 1998.

[6] A. De Luca and L. Ferrajoli, "Exploiting robot redundancy in collision detection and reaction," in *IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Nice, France, September 2008.

[7] J.-O. Kim, M. Wayne, and P. K. Khosla, "Exploiting redundancy to reduce impact force," *Journal of Intelligent and Robotic Systems*, vol. 9, no. 3, pp. 273–290, 1994.

[8] M. Howard, S. Klanke, M. Gienger, C. Goerick, and S. Vijayakumar, "Methods for learning control policies from variable-constraint demonstrations," in *From Motor Learning to Interaction Learning in Robots*, O. Sigaud and J. Peters, Eds. Springer Berlin / Heidelberg, 2010, pp. 253–291.

[9] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Trajectory formation for imitation with nonlinear dynamical systems," in *Proc. IEEE Intl Conf. on Intelligent Robots and Systems (IROS)*, 2001, pp. 752–757.

[10] S. Schaal, P. Mohajerian, and A. J. Ijspeert, "Dynamics systems vs. optimal control a unifying view," *Progress in Brain Research*, vol. 165, pp. 425–445, 2007.

[11] H. Hoffmann, P. Pastor, D. H. Park, and S. Schaal, "Biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 2009, pp. 2587–2592.

[12] S. Calinon, F. D'halluin, D. G. Caldwell, and A. G. Billard, "Handling of multiple constraints and motion alternatives in a robot programming by demonstration framework," in *Proc. IEEE-RAS Intl Conf. on Humanoid Robots (Humanoids)*, Paris, France, December 2009, pp. 582–588.

[13] S. Calinon, F. D'halluin, E. L. Sauser, D. G. Caldwell, and A. G. Billard, "Learning and reproduction of gestures by imitation: An approach based on hidden Markov model and Gaussian mixture regression," *IEEE Robotics and Automation Magazine*, vol. 17, no. 2, pp. 44–54, 2010.

[14] M. Khansari and A. G. Billard, "BM: An iterative method to learn stable non-linear dynamical systems with Gaussian mixture models," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Anchorage, Alaska, USA, May 2010, pp. 2381–2388.

[15] J. Kober and J. Peters, "Learning motor primitives for robotics," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, May 2009, pp. 2112–2118.

[16] J. Kober, "Reinforcement learning for motor primitives," Master's thesis, University of Stuttgart, Germany, August 2008.