

# Handling of multiple constraints and motion alternatives in a robot programming by demonstration framework

Sylvain Calinon<sup>†</sup>, Florent D’halluin<sup>‡</sup>, Darwin G. Caldwell<sup>†</sup> and Aude G. Billard<sup>‡</sup>

**Abstract**—We consider the problem of learning robust models of robot motion through demonstration. An approach based on Hidden Markov Model (HMM) and Gaussian Mixture Regression (GMR) is proposed to extract redundancies across multiple demonstrations, and build a time-independent model of a set of movements demonstrated by a human user. Two experiments are presented to validate the method, that consist of learning to hit a ball with a robotic arm, and of teaching a humanoid robot to manipulate a spoon to feed another humanoid. The experiments demonstrate that the proposed model can efficiently handle several aspects of learning by imitation. We first show that it can be utilized in an unsupervised learning manner, where the robot is autonomously organizing and encoding variants of motion from the multiple demonstrations. We then show that the approach allows to robustly generalize the observed skill by taking into account multiple constraints in task space during reproduction.

## I. INTRODUCTION

Robot Programming by Demonstration (PbD) covers methods by which a robot learns new skills through human guidance. PbD is perceived as particularly useful to humanoid robots, as these robots are deemed to work in direct collaboration with humans, and share a body structure corresponding closely to that of humans [1]. Humanoids are provided with an increasing number of sensory modalities and degrees of freedom, which also increases the complexity of robot control and robot learning. Learning control strategies for numerous degrees of freedom platforms deemed to interact in complex and variable environments, such as households, is faced with two key challenges. First, the complexity of the tasks to be learned is such that pure trial and error learning would be too slow. PbD appears thus a good approach to speed up learning by reducing the search space, while still allowing the robot to refine its model of the demonstration through trial and error [2]. Second, there should be a continuum between learning and control, so that control strategies can adapt in real time to drastic changes in the environment. The present work addresses both challenges in investigating methods by which PbD is used to learn the dynamics of demonstrated movements, and, by so doing,

provide the robot with a generic and adaptive model of control.

Most approaches to trajectory modeling estimate a time-dependent model of the trajectories, by either exploiting variants along the concept of spline decomposition [3]–[5] or through statistical encoding of the time-space dependencies [6], [7]. Such modeling methods are very effective and precise in the description of the actual trajectory, and benefit from an explicit time-precedence across the motion segments to ensure precise reproduction of the task. However, the explicit time-dependency require the use of other methods for realigning and rescaling the trajectories to handle time variations.

As an alternative, other approaches have considered modeling the intrinsic dynamics of motion [8]–[11]. Such approaches are advantageous in that the system does not depend on an explicit time variable and can be modulated to produce trajectories with similar dynamics in areas of the workspace not covered during training. *Hidden Markov Model* (HMM) has previously been reported as a robust probabilistic method to deal with the spatial and temporal variabilities of human motion across various demonstrations [10], [11]. To reproduce human movement, the HMM approaches proposed sofar however required either a high number of states (i.e. higher than for recognition purpose), or an additional smoothing process whose drawback is to cut down important peaks in the motion.

In this paper, we exploit the strength of two parametric statistical techniques to learn a dynamic model of a set of movements in task space.<sup>1</sup> The proposed model relies on HMM to encode the motion and on *Gaussian Mixture Regression* (GMR) [12] to robustly generalize the motion during reproduction. In comparison to other statistical regression methods such as *Locally Weighted Regression* (LWR), [13], *Locally Weighted Projection Regression* (LWPR) [14] or *Gaussian Process Regression* (GPR) [9], [15], GMR does not model the regression function directly, but models a joint probability density function of the data and then derives the regression function from the density model. For an exhaustive review of regression approaches in PbD, the interested readers can refer to [1].

In the context of robot learning by imitation, GMR offers several advantages. First, it allows us to deal with recognition and reproduction issues in a common probabilistic framework. Then, the learning process is distinct from the

<sup>†</sup>S. Calinon and D. Caldwell are with the Advanced Robotics Dept, Italian Institute of Technology (IIT), 16163 Genova, Italy. Contact: name.surname@iit.it.

<sup>‡</sup>F. D’halluin and A. Billard are with the Learning Algorithms and Systems Laboratory (LASA), Ecole Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland. Contact: name.surname@epfl.ch.

This work was supported in part by the FEELIX GROWING European project (<http://www.feelix-growing.org>) under contract FP6 IST-045169.

<sup>1</sup>The motion is planned in Cartesian space and subsequently converted to motor commands, where the tracking is assumed to be accurate.

retrieval process. A global *Expectation-Maximization* (EM) algorithm can thus be used to learn the demonstrated skill through joint density estimation during the phases of the interaction that do not require real-time computation (i.e. after the demonstrations). A faster regression process is then used for online control of the robot during reproduction.<sup>2</sup> Another advantage is that the input and output components are not pre-specified, and can thus be interchanged during reproduction. This can be advantageous for handling various types of missing data (i.e., by simultaneously considering different input/output mappings).

We showed in previous work that GMR could be used to incrementally learn a skill (i.e. without having to keep each demonstration in memory) [16], and that constraints in joint space and task space could be simultaneously considered [6]. Muehlig *et al* [7] demonstrated that the approach could be combined with optimal control methods, where the continuous and probabilistic form of GMR can be efficiently used with a gradient-based trajectory optimizer. In the papers mentioned above, time was considered as an explicit variable. We extend here the use of GMR to a dynamical system approach to get rid of the explicit time dependency. We also extend the approach to an HMM-based representation of the skill, and show that the resulting framework can be efficiently used in an unsupervised learning mode.

## II. PROBABILISTIC MODEL

Multiple examples of a skill are demonstrated to the robot in slightly different situations, where a set of positions  $x \in \mathbb{R}^{D \times M \times T}$  and velocities  $\dot{x} \in \mathbb{R}^{D \times M \times T}$  are collected during the demonstrations ( $D$  is the dimensionality of the variable  $x$ ,  $M$  is the number of demonstrations, and  $T$  is the length of a demonstration). The dataset is composed of a set of datapoints  $\{x, \dot{x}\}$ , where the joint distribution  $\mathcal{P}(x, \dot{x})$  is encoded in a continuous *Hidden Markov Model* (HMM) of  $K$  states. The output distribution of each state is represented by a Gaussian locally encoding variation and correlation information. The parameters of the HMM are defined by  $\{\Pi, a, \mu, \Sigma\}$  and learned through *Baum-Welch* algorithm [17], which is a variant of *Expectation-Maximization* (EM) algorithm.  $\Pi_i$  is the initial probability of being in state  $i$ ,  $a_{ij}$  is the transitional probability from state  $i$  to state  $j$ .  $\mu_i$  and  $\Sigma_i$  represent the center and the covariance matrix of the  $i$ -th Gaussian distribution of the HMM. Input and output components in each state of the HMM are defined as

$$\mu_i = \begin{bmatrix} \mu_i^x \\ \mu_i^{\dot{x}} \end{bmatrix} \quad \text{and} \quad \Sigma_i = \begin{bmatrix} \Sigma_i^{xx} & \Sigma_i^{x\dot{x}} \\ \Sigma_i^{\dot{x}x} & \Sigma_i^{\dot{x}\dot{x}} \end{bmatrix},$$

where the indices  $x$  and  $\dot{x}$  refer respectively to position and velocity components.

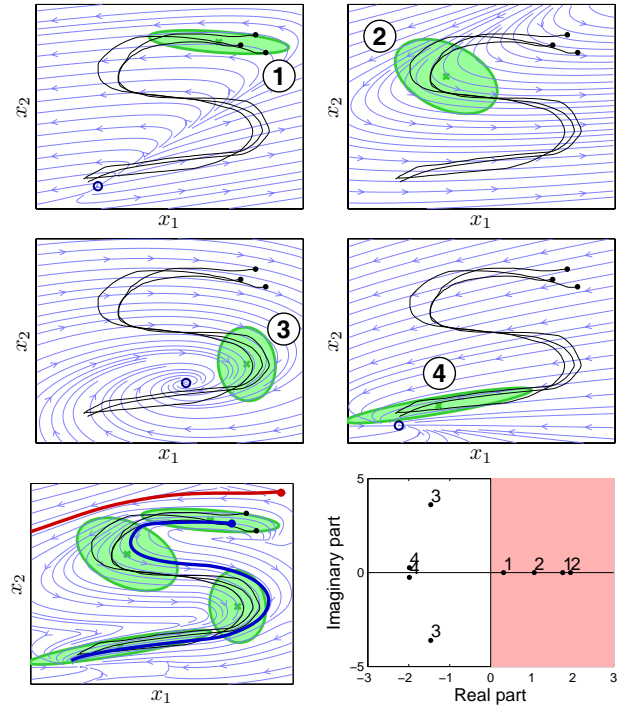


Fig. 1. Example of motion encoding and reproduction using the basic control model.

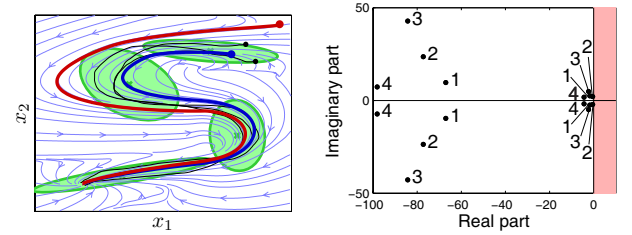


Fig. 2. Motion reproduction using the extended control model.

### A. Control model - Basic version

In the basic control model, a desired velocity  $\hat{\dot{x}}$  is estimated through *Gaussian Mixture Regression* (GMR) as

$$\hat{\dot{x}} = \sum_{i=1}^K h_i [\mu_i^{\dot{x}} + \Sigma_i^{\dot{x}x} (\Sigma_i^x)^{-1} (x - \mu_i^x)], \quad (1)$$

which is used at each iteration to control the system by estimating a new velocity, given the current position (see [18] for details).

In the GMR framework, the influence of each Gaussian is represented by a weight  $h_i \in [0, 1]$ , originally defined as the probability of an observed input to belong to the Gaussian [12]

$$h_i(x) = \mathcal{N}(x; \mu_i^x, \Sigma_i^x),$$

and normalized such that  $\sum_{i=1}^K h_i = 1$ .

<sup>2</sup>In the two experiments presented in this paper, the computation time for the learning process is below 1 second, and the computation time for each iteration of the reproduction process is below 1 millisecond.

We propose to extend this process to the estimation of a weight through an HMM representation, thus taking into consideration not only the spatial information but also the sequential information probabilistically encapsulated in the HMM

$$h_{i,t}(x) = \left( \sum_{j=1}^K h_{j,t-1} a_{ji} \right) \mathcal{N}(x; \mu_i^x, \Sigma_i^x),$$

and normalizing such that  $\sum_i^K h_{i,t} = 1$ . Here,  $h_{i,t}$  represents the *forward* variable [17], which corresponds to the probability of observing the partial sequence  $\{x_1, x_2, \dots, x_t\}$  and of being in state  $i$  at time  $t$ .

At a given instant, the regression process described in (1) can be rewritten as a mixture of linear systems

$$\dot{x} = \sum_{i=1}^K h_i (A_i' x + b_i') \text{ with } \begin{cases} A_i' = \Sigma_i^{\dot{x}x} (\Sigma_i^x)^{-1}, \\ b_i' = \mu_i^{\dot{x}} - \Sigma_i^{\dot{x}x} (\Sigma_i^x)^{-1} \mu_i^x. \end{cases} \quad (2)$$

Fig. 1 presents an example of encoding and reproduction using this control scheme, where the number of states in the HMM has been deliberately fixed to a low value. The first four graphs show the dynamic behavior of the system when using each Gaussian separately, where the circles represent the equilibrium points defined by  $-A_i'^{-1} b_i'$ . The bottom-left graph shows results for two reproduction attempts represented by blue and red thick lines, where the initial positions are represented by points. The last graph shows the poles of the system, given by the eigenvalues of matrices  $A_i'$  in (2). We observe in the first four graphs that each Gaussian representing the local distribution of  $\{x, \dot{x}\}$  can retrieve various forms of dynamics (e.g. spirals, saddle points). The last graph shows that for the first two states, the poles have a real positive part, which leads to asymptotically unstable systems when used separately. However, the first two states also bring the robot to asymptotically stable states after a few iterations. By using the basic control method, the motion is thus correctly reproduced when starting in regions that have been covered during the demonstrations (reproduction represented with blue lines). The system may however provide poor solution when initiating the motion in a region that has not been covered yet (reproduction represented by red lines).

### B. Control model - Extended version

For the reason mentioned above, we extended the basic control model to an acceleration-based controller similar to a mass-spring-damper system, where the model of the demonstrated trajectories acts as an attractor.<sup>3</sup> A target velocity  $\hat{x}$  and target position  $\hat{x}$  are first estimated at each time step through GMR. Tracking of the desired velocity  $\hat{x}$  and desired position  $\hat{x}$  is then insured by the proportional-derivative controller. The acceleration command is determined by<sup>4</sup>

$$\ddot{x} = (\hat{x} - \dot{x})\kappa^{\mathcal{V}} + (\hat{x} - x)\kappa^{\mathcal{P}}, \quad (3)$$

<sup>3</sup>Sourcecode of the algorithm is available online [19].

<sup>4</sup>In the experiments presented here, velocity and position are updated at each iteration through Euler numerical integration.

where  $\kappa^{\mathcal{V}}$  and  $\kappa^{\mathcal{P}}$  are gain parameters similar to damping and stiffness factors.

The first part of the above equation allows the robot to follow the demonstrated velocity profile.<sup>5</sup> The second part of the equation prevents the robot to depart from a known situation, and forces it to come back to this observed subspace if a perturbation occurs. By using both terms concurrently, the robot follows the learned non-linear dynamics while tracking the movement.

By rewriting  $\hat{x}$  and  $\hat{x}$  as a mixture of linear systems

$$\begin{aligned} \hat{x} &= \sum_{i=1}^K h_i (M_i x + v_i) \\ \hat{x} &= \sum_{i=1}^K h_i (M_i' \dot{x} + v_i') \end{aligned} \text{ with } \begin{cases} M_i = \Sigma_i^{\dot{x}x} (\Sigma_i^x)^{-1}, \\ M_i' = \Sigma_i^{x\dot{x}} (\Sigma_i^{\dot{x}})^{-1}, \\ v_i = \mu_i^{\dot{x}} - \Sigma_i^{\dot{x}x} (\Sigma_i^x)^{-1} \mu_i^x, \\ v_i' = \mu_i^x - \Sigma_i^{x\dot{x}} (\Sigma_i^{\dot{x}})^{-1} \mu_i^{\dot{x}}, \end{cases}$$

and knowing that  $\sum_{i=1}^K h_i = 1$ , (3) can be rewritten as

$$\ddot{x} = \sum_{i=1}^K h_i (C_i \ddot{x} + C_i' \dot{x} + C_i'') \text{ with } \begin{cases} C_i = \kappa^{\mathcal{P}} M_i' - \kappa^{\mathcal{V}} I, \\ C_i' = \kappa^{\mathcal{V}} M_i - \kappa^{\mathcal{P}} I, \\ C_i'' = \kappa^{\mathcal{V}} v_i + \kappa^{\mathcal{P}} v_i'. \end{cases}$$

The corresponding state-space representation for each Gaussian  $i$  can then be written as

$$\underbrace{\frac{d}{dt} \begin{bmatrix} x \\ \dot{x} \end{bmatrix}}_{\dot{X}} = \underbrace{\begin{bmatrix} 0 & I \\ C_i' & C_i \end{bmatrix}}_{A_i} \underbrace{\begin{bmatrix} x \\ \dot{x} \end{bmatrix}}_X + \underbrace{\begin{bmatrix} 0 \\ C_i'' \end{bmatrix}}_{b_i}, \quad (4)$$

where  $I$  is the identity matrix, and 0 represents a null matrix or vector.

Fig. 2 presents results for the same dataset and HMM as in Fig. 1, but where the extended control scheme is utilized for reproduction. By comparing the results with Fig. 1, we see in the first four plots the influence of the trajectory-attractor component combined with the trajectory-following component. The bottom-left graph shows the two reproduction attempts where the robot smoothly comes back to the motion while pursuing the demonstrated dynamics when starting from a different initial situation. The last graph shows the poles of the linear systems  $\dot{X} = A_i X + b_i$  defined in (4), consisting of four poles per Gaussian instead of two, since the system is now based on an acceleration command.

We define here the velocity gain  $\kappa^{\mathcal{V}}$  in (3) such that the model reproduces the motion with the same velocity profiles as the ones demonstrated in similar situations. The position gain  $\kappa^{\mathcal{P}}$  can be set such that it acts as a sufficient attractor to the trajectory in case of perturbation or if the system needs to start from novel situations that have not been demonstrated. It should also not be too high to avoid that the system comes back to the trajectory and stops instead of following the remainder of the motion. We thus define  $\kappa^{\mathcal{P}}$  as an adaptive gain that grows rapidly as the system departs from the area covered by the demonstrations, and is null when the system is close to the demonstrations. We thus define here  $\kappa^{\mathcal{V}}$  and  $\kappa^{\mathcal{P}}$  as

$$\kappa^{\mathcal{V}} = \frac{1}{\Delta t}, \quad \kappa^{\mathcal{P}}(x) = \kappa_{\max}^{\mathcal{P}} \frac{\mathcal{L}_{\max} - \mathcal{L}(x)}{\mathcal{L}_{\max} - \mathcal{L}_{\min}}, \quad (5)$$

<sup>5</sup>By setting  $\kappa^{\mathcal{V}} = \frac{1}{\Delta t}$  and  $\kappa^{\mathcal{P}} = 0$ , the controller is similar to (1).

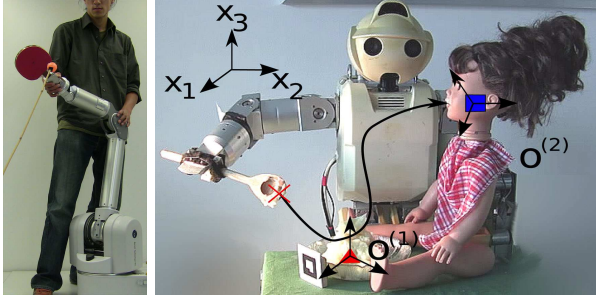


Fig. 3. Experimental setups. *Left*: Teaching the Barrett WAM robotic arm to hit a ball. *Right*: Teaching the HOAP-3 humanoid robot to feed a Robota robotic doll.

$$\text{where } \mathcal{L}_{\max} = \max_{i \in \{1, K\}} \log(\mathcal{N}(\mu_i^x; \mu_i^x, \Sigma_i^x)),$$

$$\mathcal{L}_{\min} = \min_{\substack{i \in \{1, K\} \\ x \in \mathcal{W}}} \log(\mathcal{N}(x; \mu_i^x, \Sigma_i^x)).$$

In the above equation, the notation  $\mathcal{L}$  is used to define log-likelihoods (that correspond to weighted distance measures).  $\kappa_{\max}^{\mathcal{P}}$  is the maximum gain to attain a target position.  $\kappa_{\max}^{\mathcal{P}} = 2000$  has been fixed empirically by verifying that the poles in (4) lie in the closed left half of the complex plane after having learned the HMM parameters.  $\mathcal{W}$  defines the robot’s workspace or a predetermined range of situations fixed during reproduction.  $\Delta t$  is the duration of an iteration step (a constant  $\Delta t = 0.01$  is considered here). At each iteration,  $\kappa^{\mathcal{P}}(x)$  is thus close to zero if  $x$  is within the boundary determined by the Gaussian distributions. In this situation, the controller reproduces a motion with a velocity similar to those in the demonstration sequences. On the other hand, if  $x$  is far away from the positions that have been demonstrated, the system comes back towards the closest Gaussians with a maximum gain of  $\kappa_{\max}^{\mathcal{P}}$ , still following the trend of motion in this region (determined by  $\hat{x}$ ).

Parts of the movement where a strong inconsistency has been observed indicate that the position does not need to be tracked very precisely, allowing the controller to focus on the other constraints of the task, such as following the desired velocity profile. On the other hand, portions of the movement showing strong position invariance across the multiple demonstrations will be tracked more precisely, i.e. the gain controlling the error on position will automatically be increased.

We next present two applications using this extended approach on a robotic arm and humanoid robots.

### III. EXPERIMENT WITH ROBOTIC ARM

#### A. Experimental setup

The experiment consists of learning and reproducing the motion of hitting a ball with a table tennis racket by using a Barrett WAM 7 DOFs robotic arm, see Fig. 3 *left*. One objective is to demonstrate that such movements can be transferred using the proposed approach, where the skill requires that the target is reached with a given velocity, direction and amplitude. In the experiment presented here,

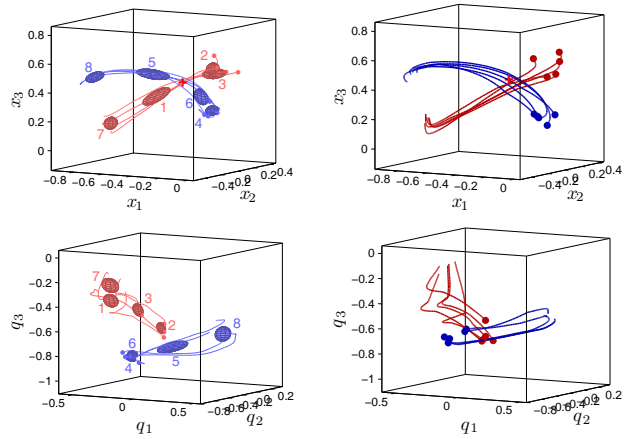


Fig. 4. Encoding and reproduction results of the table tennis experiment (in position space). *Left*: Demonstrated movements and associated Hidden Markov Model, where 8 Gaussians are used to encode the two categories of movements (the learned transitions are represented in Fig. 5). The position of the ball is depicted by a plus sign, and the initial points of the trajectories are depicted by points. The trajectories corresponding to *topspin* and *drive* strokes are respectively represented in blue and red for visualization purposes, but the robot does not have this information and is also not aware of the number of categories that has been demonstrated. *Right*: 10 reproduction attempts by starting from new random positions close to the areas where either *topspin* and *drive* strokes have been demonstrated.

we extend the difficulty of the tennis task described in [8] by assuming that the robot must hit the ball with a desired velocity retrieved from the demonstrations. The robot thus hits the ball, continues its motion and stops, which is more natural than reaching it with zero velocity.

In table tennis, *topspin* occurs when the top of the ball is going in the same direction as the ball is moving. Topspin causes the ball to drop faster than by gravity alone, and is used by players to allow the ball to be hit harder but still land on the table. The stroke with no spin is referred to as *drive*. The motion and orientation of the racket at the impact thus differ when performing a *topspin* or a *drive* stroke. Training was done by an intermediate-level player demonstrating several *topspin* and *drive* strokes to the robot by putting it in an active gravity compensation control mode, which allows the user to move the robot manually. Through this *kinesthetic teaching* process, the user *molds* the robot behavior by putting it through the task of hitting the ball with a desired spin. The ball is fixed on a stick during demonstration, and its initial position is tracked by a stereoscopic vision system.

The recordings are performed in Cartesian space by considering the position  $x$  and orientation  $q$  of the racket with respect to the ball, with associated velocities  $\dot{x}$  and  $\dot{q}$ . A quaternion representation of the orientation is used, where three of the four quaternion components are used (the fourth quaternion component is reconstructed afterwards). The user demonstrates in total 4 *topspin* strokes and 4 *drive* strokes in random order. The categories of strokes are not provided to the robot, and the number of states in the HMM is selected through *Bayesian Information Criterion* (BIC) [20]. A damped least square inverse kinematics solution with

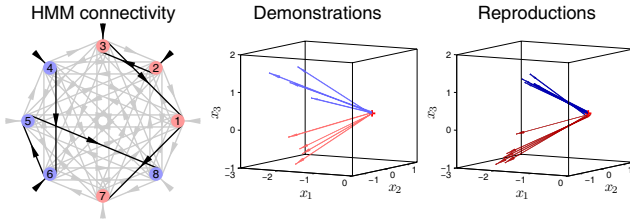


Fig. 5. *Left*: HMM representation of the transitions and initial state probabilities (the corresponding state output distributions are represented in Fig. 4). The states of the HMM are spatially organized around a circle for visualization purpose. The possible transitions are depicted inside the circle by arrows, while the probabilities of starting from an initial state are represented outside the circle by arrows. Probabilities above 0.1 are represented by black lines (self transitions probabilities are not represented here). From this representation, two different sequences defined by states transitions 2-3-1-7 and 4-6-5-8 appear, initiated by 2 or 3 for the first one, and by 4 or 6 for the second one. *Center and right*: Position and velocity of the racket at the time of the impact for the 8 demonstrations (*center*) and for the 10 reproduction attempts (*right*).

TABLE I

POSITION AND VELOCITY OF THE RACKET AT THE TIME OF THE IMPACT FOR DEMONSTRATIONS AND REPRODUCTIONS.

		Demonstrations		Reproductions	
		position [m]	velocity [m/s]	position [m]	velocity [m/s]
Drive	x	$-0.43 \pm 0.01$	$-2.19 \pm 0.04$	$-0.41 \pm 0.01$	$-2.37 \pm 0.12$
	y	$0.17 \pm 0.01$	$0.27 \pm 0.30$	$0.17 \pm 0.06$	$0.20 \pm 0.27$
	z	$0.43 \pm 0.01$	$-1.19 \pm 0.31$	$0.39 \pm 0.02$	$-1.24 \pm 0.32$
Topspin	x	$-0.43 \pm 0.01$	$-2.12 \pm 0.38$	$-0.44 \pm 0.01$	$-1.66 \pm 0.12$
	y	$0.13 \pm 0.02$	$-0.03 \pm 0.19$	$0.12 \pm 0.04$	$0.17 \pm 0.18$
	z	$0.44 \pm 0.02$	$0.73 \pm 0.38$	$0.43 \pm 0.03$	$0.73 \pm 0.16$

optimization in the null space of the Jacobian matrix is used to reproduce the task, see [6] for details.

## B. Experimental results

Fig. 4 presents the encoding and reproduction results. We see that the HMM approach reproduces an appropriate motion in the two situations. Fig. 5 *left*, presents the states transitions learned by the HMM. We see that the model has correctly learned that two different dynamics can be achieved depending on the initial position of the robot. It is thus possible to encode several motion alternatives in a single model without providing labels during demonstration, and without having to fix the number of alternatives beforehand. Adequate actions are then automatically retrieved by the robot depending on the initial situation.

Fig. 5 and Table I present the results of the strokes at the time of the impact with the ball. We see that the system correctly attains the ball at a velocity similar to the one demonstrated (in terms of both amplitude and direction). A video of the experiment is available online [19].

## IV. EXPERIMENT WITH HUMANOID ROBOTS

### A. Experimental setup

In the previous experiment, we learned trajectories in the frame of reference of a single object (the ball). This experiment with two humanoid robots considers trajectories with respect to multiple landmarks. A *HOAP-3* humanoid robot from *Fujitsu* is used in the experiment. It has in total

28 DOFs, of which the 8 DOFs of the upper torso are used in the experiment (4 DOFs per arm). A *kinesthetic teaching* process is used for demonstration. The selected motors are set to passive mode, which allows the user to move freely the corresponding degrees of freedom while the robot executes the task. The kinematics of each joint motion are recorded at a rate of 1 kHz.

The experiment consists of feeding a *Robota* robotic doll [21], where *HOAP-3* first brings a spoon to a plate of mashed potatoes and then moves it towards *Robota*'s mouth, see Fig. 3 *right*. Four demonstrations are provided by changing the initial positions of the landmarks from one demonstration to the other. The experimenter explicitly signals the start and the end of the demonstrations to the robot.

The set of landmarks (or objects) tracked by the robot is pre-defined. The position of the plate is recorded through a patch attached to it, which is tracked by an external vision system placed to the side of the robot. The position of *Robota*'s mouth is tracked by proprioception through the robot's motor encoders. *HOAP-3*'s left arm is rigidly attached to *Robota* and *HOAP-3* is connected to *Robota*'s head encoders. *Robota*'s head is thus considered as an additional link to the kinematic model of the robot. This allows to precisely track the position of the mouth during demonstration and reproduction, without the use of a visual marker that would easily be occluded by the spoon moving close to the mouth.

In the demonstration phase, the position  $x$  of the end-effector is collected in the frame of reference of the robot's torso (fixed frame of reference as the robot is seated during the experiment). This trajectory is expressed in the frames of reference of the different landmarks (moving frames of references) defined for each landmark  $n$  by position  $o^{(n)}$  and orientation matrix  $R^{(n)}$

$$x^{(n)} = R^{(n)\top} (x - o^{(n)}), \quad \dot{x}^{(n)} = R^{(n)\top} \dot{x}.$$

Encoding in a *Hidden Markov Model* is computed for each landmark as described in Section II. During the reproduction phase, for new position  $o'^{(n)}$  and orientation  $R'^{(n)}$  of the landmarks, the generalized position  $\hat{x}$  and velocity  $\hat{\dot{x}}$  of the end-effector with respect to the different landmarks is projected back to the frame of reference attached to the torso

$$\hat{x}'^{(n)} = R'^{(n)} \hat{x}^{(n)} + o'^{(n)}, \quad \hat{\dot{x}}'^{(n)} = R'^{(n)} \hat{\dot{x}}^{(n)}.$$

The associated covariances matrices are transformed through the linear transformation property of Gaussian distributions<sup>6</sup>

$$\hat{\Sigma}'^{x(n)} = R'^{(n)} \hat{\Sigma}^{x(n)} R'^{(n)\top}, \quad \hat{\Sigma}'^{\dot{x}(n)} = R'^{(n)} \hat{\Sigma}^{\dot{x}(n)} R'^{(n)\top}.$$

At each time step, the command defined in (3) is used to retrieve the desired velocity  $\hat{x}'$  and desired position  $\hat{x}'$ , where the resulting distributions  $\mathcal{N}(\hat{x}', \hat{\Sigma}'^{x'})$  and  $\mathcal{N}(\hat{\dot{x}}', \hat{\Sigma}'^{\dot{x}'})$  are respectively computed through the Gaussian products  $\prod_{n=1}^N \mathcal{N}(\hat{x}'^{(n)}, \hat{\Sigma}'^{x(n)})$  and  $\prod_{n=1}^N \mathcal{N}(\hat{\dot{x}}'^{(n)}, \hat{\Sigma}'^{\dot{x}(n)})$ . This allows the system to combine automatically the different constraints associated with the landmarks.

<sup>6</sup>Covariance matrices  $\hat{\Sigma}^{x(n)}$  and  $\hat{\Sigma}^{\dot{x}(n)}$  are estimated through GMR, see [6] for details.

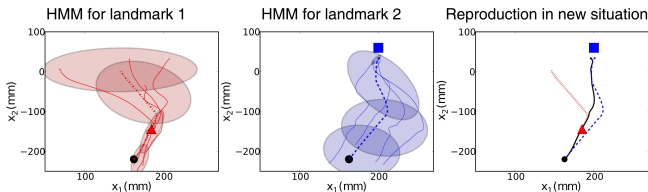


Fig. 6. Encoding and reproduction results, where relative trajectories with respect to the two landmarks are encoded in two HMMs of 4 states. Each Gaussian encodes position and velocities information along the task. Generated trajectories using the corresponding models are represented with dashed lines, where the dots show the initial positions. The position of the landmarks are represented with a triangle for the plate and with a square for Robot’s mouth. *Right*: Reproduction results where the final reproduction is represented by a solid line. The reproduction shows that the robot tends to satisfy the first constraint first (to reach for the plate), and then switches smoothly to the second constraint (to reach for Robot’s mouth).

## B. Experimental results

Fig. 6 presents the encoding results. The *left* and *center* graphs highlight through the forms of the Gaussians that parts of the motion are more constrained than others. With respect to landmark 1, strong consistency among the demonstrations have been observed at the beginning of the gesture (motion of the spoon in the mashed potatoes), which is reflected by the narrower form of the ellipses at the beginning of the motion. With respect to landmark 2, strong consistency among the demonstrations have been observed at the end of the gesture (when reaching for Robot’s mouth). Fig. 6 *right*, presents the reproduction results. We see that the robot automatically combines the two sets of constraints (actions associated with the plate and with Robot’s mouth) to find a trade-off satisfying probabilistically the constraints observed during the demonstrations. A video of the experiment is available online [19].

## V. DISCUSSION

The proposed system shares similarities with the *Dynamic Movement Primitives* (DMP) approach proposed by Ijspeert *et al* [8], where a set of *Vector Integration To Endpoint* (VITE) primitives is used to reach a desired target, and where an additional term is used to anchor that the velocity vanishes at the end of the movement (phase variable acting as a decay term to ensure that the system asymptotically converges to a reaching point or limit cycle. By using a formulation similar to the one used in Sec. II (see also the reformulation in [22]), DMP can be computed as

$$\ddot{x} = (\hat{x} - x) \kappa^p - \dot{x} \kappa^v, \quad \text{with} \quad \hat{x} = \sum_{i=1}^K h_i \mu_i^x, \quad (6)$$

where gains  $\kappa^p$  and  $\kappa^v$  are fixed to obtain a critically damped system. Centers  $\mu_i^x$  are learned through regression from the observed data, either incrementally or in a batch mode (e.g. through least-squares regression). The weights  $h_i$  are defined by Gaussian distributions

$$h_i(s) = \mathcal{N}(s; \mu_i^s, \Sigma_i^s),$$

and normalized such that  $\sum_i^K h_i = 1$ .  $s \in [0, 1]$  is a decay term initialized with  $s = 1$  and converging to zero through

a canonical system<sup>7</sup>  $\dot{s} = -\alpha s$ . Centers  $\mu_i^s$  are equally distributed between 1 and 0, and variance parameters  $\Sigma_i^s$  are set to a constant value depending on the number of kernels (here,  $\alpha = 0.1$ ,  $\kappa^p = (\kappa^v)^2/4$  and  $\Sigma_i^s = \frac{3}{2\kappa^v}$ ). By determining the weights through the decay term  $s$ , the system is thus guaranteed to converge to the last attractor  $\mu_K^x$ .

One drawback is that a heuristic must be explicitly defined to let the system recompute the value of  $s$  in case of perturbation. For example, if the robot needs to reproduce only one part of the motion, or if the target is moving, the canonical system must be redefined to re-estimate the canonical variable  $s$  in consequence. In the proposed HMM/GMR approach, the weights  $h_i$  are inherently encapsulated in the model (HMM representation of the observed movement) and do not require to have an explicit parametrization of a temporal decay. This has several advantages: (i) it allows spatial and temporal distortions to be handled very flexibly; (ii) training and refinement of a skill can be performed by providing partial demonstrations; and (iii) reaching and cyclic motions can be learned without having to fix the form of the dynamics in advance.

As demonstrated by the two experiments, the proposed HMM/GMR approach is thus not constrained to movements with a unique zero-velocity attractor point (e.g. several points of interest to be attained with a desired velocity can be automatically extracted along the motion). A single model is used to encode multivariate data, which allows automatic learning of the correlations between the different variables, and the use of this information for the reproduction process.<sup>8</sup> By considering the regression terms in (1), the covariance matrices provide local information on the spread of each center  $\mu_i$  and thus provides an efficient regression estimate based on the observed variability across the different variables along the task, even when a low number of Gaussians is considered.

One drawback of learning the weights  $h_i$  from examples concerns the stability of the system in regions that have not been covered during the demonstrations. We have seen that the extended control model can be used in such situations to help at generalizing the observed skill to new situations (see Fig. 2, and Fig. 1 for comparison). In this example, the poles with strictly negatives real values highlight asymptotic stability properties, but it is not guaranteed that the position of these stable points remain in the robot’s workspace. For reaching motion, the last state learned by the model will typically act as an attractor point and the HMM transition probabilities insure to some extent the convergence to this state. Motion behavior is however not straightforward to analyze, as stability depends on the demonstrations provided to the robot. If a reaching motion is observed, the robot

<sup>7</sup>Note that this secondary term creates an implicit time-dependency, whereby each step of motion is incremented following the clock of the canonical system.

<sup>8</sup>To handle multivariate data, DMP considers the different variables as separated processes synchronized by the phase variable, while HMM encapsulates the complete correlation information.

learns to stop at the target position. If a cyclic motion is observed, the robot follows the loop endlessly. The process is generic such that the type of motion does not need to be specified, i.e. the robot utilizes exactly the same system to handle these two dynamic behaviors. There is unfortunately a counterbalance to this generic property. If the robot observes someone moving in one direction without stopping (e.g. by providing recordings where the deceleration phase is not demonstrated), it will also follow the motion in this direction without stopping.<sup>9</sup> Future work will concentrate on directly incorporating the stability constraint in the learning process to insure that the system remains stable when extrapolating the demonstrated movements to other regions.

## VI. CONCLUSION

We proposed a dynamical systems approach to learn new skills by representing the evolution of the joint distribution of position  $x$  and velocity  $\dot{x}$  in a *Hidden Markov Model*. The reproduction process is based on an extended version of *Gaussian Mixture Regression* (GMR) that takes advantage of the statistical properties of HMM to handle both spatial and temporal variabilities. Two experiments were presented to illustrate the properties of the system. The use of the approach in an unsupervised learning manner was first highlighted, with several motion alternatives encoded in a single HMM, and without associating the different demonstrations with classes or labels. The second experiment showed that several constraints could be extracted automatically through multiple demonstrations. During reproduction, the extracted constraints were then used to find an appropriate controller for the robot.

## 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] F. Guenter, M. Hersch, S. Calinon, and A. Billard, "Reinforcement learning for imitating constrained reaching movements," *Advanced Robotics*, vol. 21, no. 13, pp. 1521–1544, 2007.
- [3] A. Ude, "Trajectory generation from noisy positions of object features for teaching robot paths," *Robotics and Autonomous Systems*, vol. 11, no. 2, pp. 113–127, 1993.
- [4] H. Friedrich, S. Muench, R. Dillmann, S. Bocionek, and M. Sassin, "Robot programming by demonstration (RPD): Supporting the induction by human interaction," *Machine Learning*, vol. 23, no. 2, pp. 163–189, 1996.
- [5] J. Aleotti and S. Caselli, "Robust trajectory learning and approximation for robot programming by demonstration," *Robotics and Autonomous Systems*, vol. 54, no. 5, pp. 409–413, 2006.
- [6] S. Calinon and A. Billard, "Learning a skill by imitation through gaussian mixture regression - handling of constraints in joint space and task space," 2009, in Press.
- [7] M. Muehlig, M. Gienger, S. Hellbach, J. Steil, and C. Goerick, "Task-level imitation learning using variance-based movement optimization," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Kobe, Japan, May 2009.
- [8] A. 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)*, Maui, USA, 2001, pp. 752–757.
- [9] D. Grimes, R. Chalodhorn, and R. Rao, "Dynamic imitation in a humanoid robot through nonparametric probabilistic inference," in *Proc. Robotics: Science and Systems (RSS)*, Philadelphia, PA, USA, August 2006, pp. 1–8.
- [10] T. Inamura, I. Toshima, and Y. Nakamura, "Acquiring motion elements for bidirectional computation of motion recognition and generation," in *Experimental Robotics VIII*, B. Siciliano and P. Dario, Eds. Springer-Verlag, 2003, vol. 5, pp. 372–381.
- [11] D. Kulic, W. Takano, and Y. Nakamura, "Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden markov chains," *Intl Journal of Robotics Research*, vol. 27, no. 7, pp. 761–784, 2008.
- [12] Z. Ghahramani and M. Jordan, "Supervised learning from incomplete data via an EM approach," in *Advances in Neural Information Processing Systems*, J. D. Cowan, G. Tesauero, and J. Alspector, Eds., vol. 6. Morgan Kaufmann Publishers, Inc., 1994, pp. 120–127.
- [13] S. Schaal and C. Atkeson, "Constructive incremental learning from only local information," *Neural Computation*, vol. 10, no. 8, pp. 2047–2084, 1998.
- [14] S. Vijayakumar, A. D'souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Computation*, vol. 17, no. 12, pp. 2602–2634, 2005.
- [15] D. Nguyen-Tuong and J. Peters, "Local gaussian process regression for real-time model-based robot control," in *IEEE/RSSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Nice, France, September 2008, pp. 380–385.
- [16] S. Calinon and A. Billard, "Incremental learning of gestures by imitation in a humanoid robot," in *Proc. ACM/IEEE Intl Conf. on Human-Robot Interaction (HRI)*, Arlington, VA, USA, March 2007, pp. 255–262.
- [17] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77:2, pp. 257–285, February 1989.
- [18] M. Hersch, F. Guenter, S. Calinon, and A. Billard, "Dynamical system modulation for robot learning via kinesthetic demonstrations," *IEEE Trans. on Robotics*, vol. 24, no. 6, pp. 1463–1467, 2008.
- [19] S. Calinon, "Robot programming by demonstration: A probabilistic approach," <http://programming-by-demonstration.org>, April 2009.
- [20] G. Schwarz, "Estimating the dimension of a model," *Annals of Statistics*, vol. 6, pp. 461–464, 1978.
- [21] A. Billard, "Robota: Clever toy and educational tool," *Robotics and Autonomous Systems*, vol. 42, pp. 259–269, 2003.
- [22] H. Hoffmann, P. Pastor, D. 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)*, Kobe, Japan, May 2009, pp. 2587–2592.

<sup>9</sup>Here, a separate low-level motor controller was used to prevent the robot moving beyond joint limits.