Learning and reproduction of gestures by imitation

An approach based on Hidden Markov Model and Gaussian Mixture Regression

Sylvain Calinon, Florent D'halluin, Eric L. Sauser, Darwin G. Caldwell and Aude G. Billard

***Abstract*—We present a probabilistic approach to learning robust models of human motion through imitation. The combination of Hidden Markov Model (HMM) and Gaussian Mixture Regression (GMR) allows us to extract redundancies across multiple demonstrations and build time-independent models to reproduce the dynamics of the observed movements. The approach is first compared with state-of-the-art approaches by using generated trajectories sharing similar characteristics to those of humans. Three applications on different types of robots are then presented. An experiment with the iCub humanoid robot acquiring a bimanual dancing motion is first presented to show that the system can cope with cyclic and crossing motions. An experiment with a 7 DOFs WAM robotic arm learning the motion of hitting a ball with a table tennis racket is presented to highlight the possibility to encode several movements in a single model. Finally, an experiment with a HOAP-3 humanoid robot holding a spoon and learning to feed the Robota humanoid robot is presented. It shows the capability of the system to handle several constraints simultaneously.**

***Index Terms*—Robot programming by demonstration, Learning by imitation, Gaussian mixture regression, Hidden Markov Model.**

I. Introduction

ROBOT Programming by Demonstration (PbD) covers methods by which a robot learns new skills through human guidance. Also referred to as learning by imitation, lead-through teaching, tutelage or apprenticeship learning, PbD takes inspiration from the way humans learn new skills by imitation to develop methods by which new tasks can be transmitted to a robot [1], [2].

Learning control strategies for numerous degrees of freedom platforms that interact in complex and variable environments 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 thus appears as a way 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 [3]. Second, there should be a continuum between learning and control, so that control strategies can adapt in real time to perturbations, such as changes of position and orientation of objects. The present work addresses both challenges in investigating and comparing methods by which PbD is used to learn the dynamics of demonstrated movements, and, hence, provides the robot with a generic and adaptive model of control.

This work was supported in part by the FEELIX GROWING European project under contract FP6 IST-045169, and by the AMARSi European project under contract FP7-ICT-248311.

A. Related work and motivations

PbD is of interest for different levels of task representation. A large body of work in PbD follows a symbolic approach to the representation and encoding of the tasks, see e.g. [4]–[9]. Such a symbolic description offers the advantage that it provides a way to easily tackle sequences or hierarchies of actions. One major drawback, however, is that they rely on strong biases to predefine the important cues and to segment those efficiently.

Most approaches to trajectory modeling estimate a time-dependent model of the trajectories, by either exploiting variants along the concept of spline decomposition [10], [11] or through an explicit encoding of the time-space dependencies [12]. Such modeling methods are 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 of these models requires the use of other methods for realigning and scaling the trajectories to handle spatial and temporal perturbations. As an alternative, other approaches have considered modeling the intrinsic dynamics of motion [13]–[16]. 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. We use HMM in this work, which has previously been reported as a robust probabilistic method to handle the spatial and temporal variabilities of human motion across various demonstrations [14], [16]. Most of the approaches proposed so far, however, require either a high number of states to correctly reproduce the motion (i.e. higher than for recognition purposes), or an additional smoothing procedure which has the drawback of reducing important peaks in the motion.

The proposed model relies on *Gaussian Mixture Regression* (GMR) [12], [17], [18] to generalize the motion during reproduction. In contrast to other regression methods such as *Locally Weighted Regression* (LWR) [19], *Locally Weighted Projection Regression* (LWPR) [20], or *Gaussian Process Regression* (GPR) [15], [21], GMR does not model the regression function directly, but models a joint probability density function of the data. It then derives the regression function from the joint density model [1]. This is an advantage in some robotic applications since the input and output components are only specified at the very last step of the process. Density estimation can thus be learned in an off-line phase, while the regression process can be computed very rapidly. It can also handle different sources of missing data, as the system is able to consider any combination of input/output mappings during

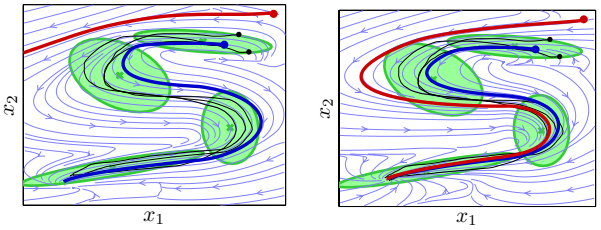

Fig. 1. Example of motion encoding and reproduction using the unstable estimate of the motion dynamics (*left*) and the stabilizer (*right*).

the retrieval phase.

II. Proposed approach

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 D-dimensional positions $x = \{x_t\}_{t=1}^{T_m}$ and velocities $\dot{x} = \{\dot{x}_t\}_{t=1}^{T_m}$. 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 using the *Baum-Welch* algorithm [22], which is a variant of the *Expectation-Maximization* (EM) algorithm. Π_i is the initial probability of being in state i, a_{ij} is the transitional probability from state i to state j. μ_i and Σ_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

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

with $i \in \{1,\dots,K\}$. The indices ^{x} and $^{\dot{x}}$ refer respectively to position and velocity.

A desired velocity $\hat{\dot{x}} = \sum_{i=1}^{K} h_i(x) P(\dot{x}|x,i)$ is estimated through *Gaussian Mixture Regression* (GMR) as

$$\hat{\dot{x}} = \sum_{i=1}^{K} h_i(x) \left[\mu_i^{\dot{x}} + \Sigma_i^{\dot{x}x}(\Sigma_i^x)^{-1}(x - \mu_i^x)\right]. \qquad (1)$$

Given the current position, a velocity command is estimated iteratively to control the system. In [23], we considered a second order model. Here, our estimate is done solely on the first derivative, which in practice proves to be more robust to compute. In the original GMR framework [17], the influence of the different Gaussians is represented by weights $h_i \in \mathbb{R}_{[0,1]}$, defined as the probability of an observed input belonging to each of the Gaussians. We propose to extend this estimation by recursively computing a likelihood through the HMM representation, thus taking into consideration not only the spatial information but also the sequential information probabilistically encapsulated in the HMM[1]

$$h_i(x_t) = \frac{\left(\sum_{j=1}^{K} h_j(x_{t-1})\, a_{ji}\right) \mathcal{N}(x_t;\ \mu_i^x, \Sigma_i^x)}{\sum_{k=1}^{K}\left[\left(\sum_{j=1}^{K} h_j(x_{t-1})\, a_{jk}\right) \mathcal{N}(x_t;\ \mu_k^x, \Sigma_k^x)\right]}.$$

[1]We will omit the indices t in further equations.

Here, $h_i(x_t)$ represents the HMM *forward* variable [22], initialized with $h_i(x_1) = \frac{\pi_i \mathcal{N}(x_1;\, \mu_i^x, \Sigma_i^x)}{\sum_{k=1}^{K}\left[\pi_k \mathcal{N}(x_1;\, \mu_k^x, \Sigma_k^x)\right]}$, and corresponding to the probability of observing the partial sequence $\{x_1, x_2, \dots, x_t\}$ and of being in state i at time t.

Fig. 1 *left* presents an example of encoding and reproduction using this basic control scheme, where the number of states in the HMM has been deliberately fixed to a low value. Two reproduction attempts are represented by the thick blue and red lines, where the initial positions are represented by points. When the motion is initialized nearby the original demonstrations, the system behaves as desired. However, if initialized in a region that has not been covered during the demonstrations (see trajectory represented by red lines), the system does not follow the desired trajectory.

To tackle the inherent instabilities of the model of Eq. (1), we add a secondary term that takes the form of a mass-spring-damper system that brings back the trajectories toward the centers of the Gaussians. Each of these centers acts as a "transient tracking point" to this secondary system, hence driving the motion along the way. Transition across tracking points is ensured by the transition probabilities of the HMM.

The stabilizer is derived as follows. At each time step, a target velocity and target position are retrieved from our estimate of the dynamics of motion, following Eq. (1) and $\hat{x} = \sum_{i=1}^{K} h_i(x)[\mu_i^x + \Sigma_i^{x\dot{x}}(\Sigma_i^{\dot{x}})^{-1}(\dot{x} - \mu_i^{\dot{x}})]$.

Tracking of the desired velocity $\hat{\dot{x}}$ and desired position $\hat{x}$ is then ensured by the proportional-derivative controller. The acceleration command is determined by

$$\ddot{x} = \overbrace{(\hat{\dot{x}} - \dot{x})\kappa^{\mathcal{V}}}^{\ddot{x}^{\mathcal{V}}} + \overbrace{(\hat{x} - x)\kappa^{\mathcal{P}}}^{\ddot{x}^{\mathcal{P}}}, \qquad (2)$$

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

In the above equation, $\ddot{x}^{\mathcal{V}}$ allows the robot to follow the demonstrated velocity profile. $\ddot{x}^{\mathcal{P}}$ prevents the robot from departing from a known situation, and forces it to come back to the subspace of demonstrations, if a perturbation occurs. The non-linear dynamics of the movement is thus approximated by a mixture of linear systems, where the influence of the different linear models is estimated through a non-linear process. Eq. (2) can be formulated as a mixture of linear systems, see [24] for details.

Note that the tracking term in (2) may distort the original estimate of the dynamics (oscillations around the original demonstrations). Avoiding such oscillations and minimizing the distortions depends on choosing carefully the gains parameters. In practice, for the experiments reported here (and for well chosen gains), the behavior of the system followed the desired dynamics. Analysis and solutions to the problem of stabilizing the first order system can be found in [25]. Fig. 1 *right* presents reproduction results with the stabilizer, where the robot smoothly comes back to the demonstrated movement when starting from a different initial situation.

Using constant gains in Eq. (2) may distort the demonstrated dynamics of the movement in-between two consecutive Gaussians (the effect tends to disappear by increasing the number of Gaussians). While this solution may be acceptable for some

tasks, we suggest here the use of adaptive gains. By setting a proportional gain that decreases when the system is close to the demonstrated trajectories, the system reproduces not only the demonstrated path, but also follows the dynamics of the movement while following this path, see [26] for details.

Parts of the movement where the variations across the demonstrations are high indicate that the position does not need to be tracked very precisely. Setting adaptive gains as in [26] allows the controller to focus on the other constraints of the task, such as following a desired velocity. On the other hand, parts of the movement exhibiting strong invariance across the demonstrations will be tracked more precisely, i.e. the gain controlling the error on position will automatically be increased.

III. Evaluation through generated data

A. Generation of human-like motion data

To analyze systematically the proposed system, several sets of trajectories are randomly generated. First, a set of keypoints X of D dimensions is created (each variable $\{X_i\}_{i=1}^{D}$ is generated with a uniform random distribution $\mathcal{U}(0,1)$).

A *Vector Integration To Endpoint* (VITE) system, which has been suggested as a biologically plausible model of human reaching movement [27], is then used to generate trajectories by starting from a first keypoint and recursively defining the next keypoint as the target. It is defined here as a critically damped mass-spring-damper controller $\ddot{x} = (X - x)\kappa^{\mathcal{P}} - \dot{x}\kappa^{\mathcal{V}}$ with parameters $\kappa^{\mathcal{V}} = 25$, $\kappa^{\mathcal{P}} = (\kappa^{\mathcal{V}})^2/4$, and time step $\tau = 0.003$ sec. Every 50 iterations, the target is switched to the next keypoint. For the last keypoint, 50 additional iterations are used to let the system converge to the last keypoint. To simulate motion variability, each dataset consists of 3 trajectories produced by slightly varying, with a Gaussian noise $\mathcal{N}(0, 0.1)$, the positions of the keypoints. The resulting trajectories present natural looking motions that share similarities with those of humans. The automation of the generation process allows us to flexibly evaluate the imitation performance of our algorithm with respect to several datasets of different dimensionalities.

B. Comparison with other approaches

The approach that we propose in this paper will be further denoted as **HMM**, as its core representation is based on a Hidden Markov Model. We compare this approach with four state-of-the-art methods that have proven good performance in robotics applications.

TGMR: *Time-dependent Gaussian Mixture Regression* [12] is based on our previous work, where time is used as an explicit input variable, and where the demonstrations are first aligned in time through *Dynamic Time Warping* (DTW). Then, the distribution of temporal and spatial variables $\{t, x, \dot{x}\}$ is encoded in a *Gaussian Mixture Model* (GMM). At each time step during the reproduction process, a desired position $\hat{x}$ and a desired velocity $\hat{\dot{x}}$ are then retrieved through GMR by estimating $P(x, \dot{x}|t)$. The controller used by the robot to reproduce the skill is the mass-spring damper system defined in Eq. (2).

LWR: *Locally Weighted Regression* [19] is a memory-based probabilistic approach. It is used here to estimate at each time step a desired position $\hat{x}$ and a desired velocity $\hat{\dot{x}}$. Each datapoint of the dataset participates in the estimation of the solution by using a Gaussian kernel with fixed diagonal covariance matrix centered at the current position to weight the influence of each datapoint. The controller used by the robot is the mass-spring damper system defined in Eq. (2).

LWPR: *Locally Weighted Projection Regression* is an incremental regression algorithm that performs piecewise linear function approximation [20]. The algorithm does not require storage of the training data and has been proved to be efficient in a variety of robot learning tasks including high dimensional data. We use here an implementation of LWPR with the input space defined by a set of receptive fields with full covariance matrices. By detecting locally redundant or irrelevant input dimensions, the method locally reduces the dimensionality of the input data by finding local projections through *Partial Least Squares* (PLS) regression. The learning parameters have been set based on the recommendations provided in [20]. During reproduction, LWPR is used at each iteration to estimate a desired velocity $\hat{\dot{x}}$ given the current position x. The receptive fields are then used to determine a desired position $\hat{x}$ in a similar manner to the methods above. The controller used by the robot is the mass-spring damper system defined in Eq. (2).

DMP: The *Dynamic Movement Primitives* approach was originally proposed by Ijspeert *et al* [13]. The method allows a target to be reached by modulating a set of mass-spring-damper systems. This allows a particular path to be followed with the guarantee that the velocity vanishes at the end of the movement. A phase variable acts as a decay term to ensure that the system asymptotically converges to a reaching point.

C. Metrics of imitation performance

Five metrics are used to evaluate a reproduction attempt $x' \in \mathbb{R}^{(D\times T)}$ with respect to the set of demonstrations $x \in \mathbb{R}^{(D\times M\times T)}$ rescaled in time with $T = \frac{\sum_{m=1}^{M} T_m}{M}$.

RMS error $\mathcal{M}_1$: This metric evaluates the generalization capability by measuring how well the reproduced trajectory matches the different demonstrations. It evaluates the accuracy of the reproduction in terms of spatial and temporal information, where a *root-mean-square* (RMS) error on position (with respect to the $M = 3$ demonstrations of the dataset) is computed along the reproduced motion $\mathcal{M}_1 = \frac{1}{MT}\sum_{m=1}^{M}\sum_{t=1}^{T}||x'_t - x_{m,t}||$.

RMS error after DTW $\mathcal{M}_2$: For this metric, the reproduced motion is first temporally aligned with the demonstrations through *Dynamic Time Warping* (DTW), and a RMS error on position similar to $\mathcal{M}_1$ is then computed. In contrast with $\mathcal{M}_1$, spatial information is prioritized here (i.e., the metric compares the path followed by the robot instead of the exact trajectory along time).

Norm of jerk $\mathcal{M}_3$: This metric evaluates the smoothness of the reproduction based on RMS jerk quantification. This measure, based on the derivative of acceleration, has been shown to be a good candidate to evaluate smoothness of human motion [28] $\mathcal{M}_3 = \frac{1}{T}\sum_{t=1}^{T}||\dddot{x}'_t||$.

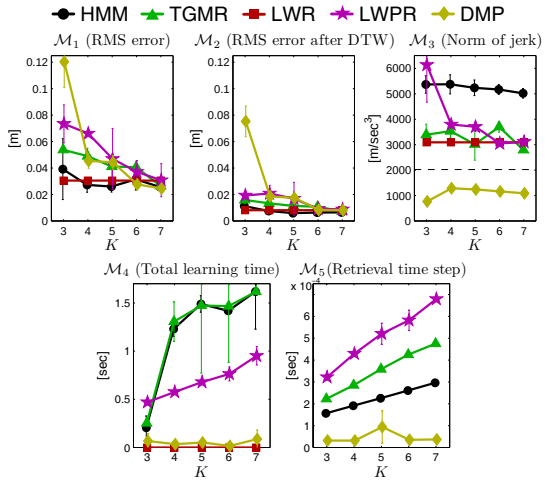

Fig. 2. Influence of the number of states K on the metrics, for $D = 7$ dimensions. The dashed line in $\mathcal{M}_3$ represents the mean RMS jerk of the demonstrations.

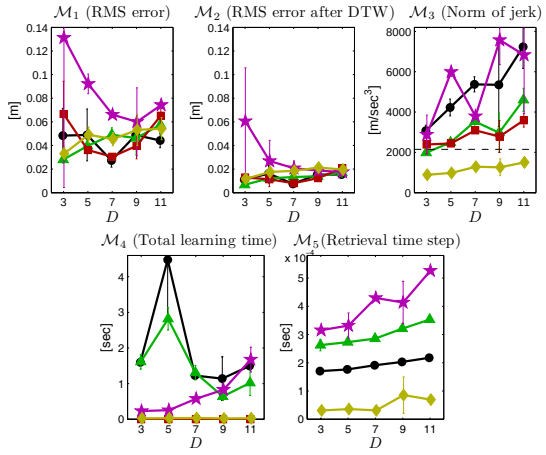

Fig. 3. Influence of the dimensionality D of the dataset on the metrics, for $K = 4$ states (see Fig. 2 for legend).

Learning time $\mathcal{M}_4$: Computation time of the learning process.

Retrieval duration $\mathcal{M}_5$: Computation time of the retrieval process for one iteration.

$\mathcal{M}_4$ and $\mathcal{M}_5$ are evaluated through non-optimized Matlab implementations of the algorithms running on a 2.5GHz Pentium processor.

D. Evaluation results

Three different sets of movements are generated with the approach presented in Sec. III-A. For each set of movements, three reproduction attempts are performed. This process is then repeated for various numbers of states, dimensionalities and ranges of perturbation. The quantitative results are presented in Figs 2-3.

Fig. 2 shows the influence of the number of states K in the model (or basis functions), for the different methods and metrics. As LWPR is an online incremental learning method, the parameter that determines when new basis functions are created (parameter w_{gen} in [20]) has been gradually increased until the number of receptive fields matches the desired number of states. We see with $\mathcal{M}_1$ and $\mathcal{M}_2$ that all methods perform very well, accurately following the demonstrated movements in terms of RMS errors. By encapsulating correlation information across input and output variables, HMM performs well with a very small number of states. We see with $\mathcal{M}_3$ that DMP reproduces the smoothest movement (it actually smoothes the original demonstrations, see RMS jerk depicted in dashed line). It is noticeable that smoothness is not much affected by the number of states in general. For $\mathcal{M}_4$, DMP and LWR show the best performance in terms of the computation time used by the learning process (LWR is zero as it is a data-driven approach without learning), while HMM and TGMR (both trained by *Expectation-Maximization*) show a slightly worse performance. For a better comparison with the online learning nature of LWPR, 10 passes have been performed with the dataset shuffled randomly. It should thus be noted that by using a single pass, the computation time can be reduced by an order of magnitude.

In this experiment, we concentrated on a case where the learning process is separated from the retrieval process. In this context, both a batch learning process and an online learning process can be employed. For the subset of robotic tasks that we consider here, the computation time needed for learning has less importance than the one required for real-time reproduction of a skill. In Fig. 2, we see that all the methods learned in less than 2 sec. We make the assumption that this idle time remains acceptable for the user. Note, however, that a stricter constraint can be considered by modifying the stopping criterions of the iterative Expectation-Maximization procedure to take into account a measure of the acceptable waiting time.

For $\mathcal{M}_5$, the computation time used by LWR for reproduction is not competitive and is thus not depicted here (it goes over 7×10^{-2} sec. as in the proposed implementation, each datapoint contributes to the estimation). The other approaches show a linear dependency on the number of states and are all suitable for online application in robotics (less than 1 millisecond per iteration for the considered number of states).

Fig. 3 shows the influence of the dimensionality D on the metrics for the different approaches (see legend in Fig. 2), when considering $K = 4$ states in the model. We see with $\mathcal{M}_1$ and $\mathcal{M}_2$ that the methods perform equally well in terms of RMS errors.

When the dimensionality is low, the difficulty is to correctly handle the crossing points that can appear when randomly generating trajectories (i.e., when passing through the same point several times during a demonstration). When the dimensionality is high, these crossings are less likely to occur. However, the difficulty is in this case to efficiently handle the sparsity of the data (curse of dimensionality). This fact is reflected by the data, and is particularly noticeable for LWR. The comparison with LWPR is not very informative here, as the lower performance is related to the online nature of the learning process. Quantitative comparison would be unfair as an online algorithm cannot determine in advance whether loops in the motion will be encountered, while a batch learning process can cluster the crossing points more easily.

For $\mathcal{M}_4$ in Fig. 3, we see that the computation time of

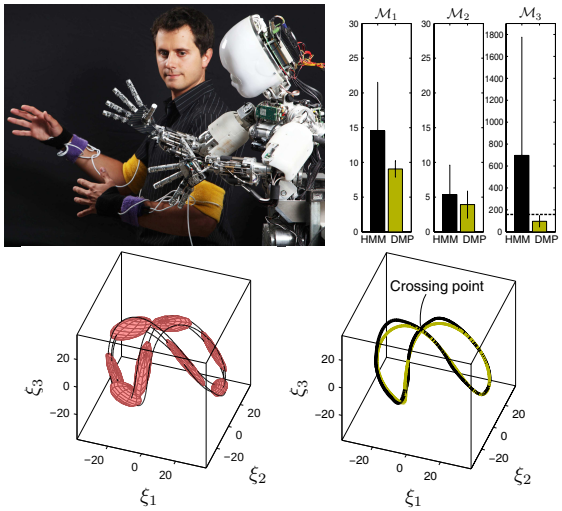

Fig. 4. *Top:* Demonstration of the skill and evaluation. The reproductions with the HMM and DMP processes are respectively represented with black and yellow lines. For $\mathcal{M}_3$, the dotted line depicts the RMS jerk value for the training data. *Bottom:* Model and reproductions. For visualization purpose, the 14 DOFs periodic trajectories and associated HMM have been projected into a latent space of 3 dimensions $\{\xi_1, \xi_2, \xi_3\}$ through *Principal Component Analysis* (PCA).

Expectation-Maximization (EM) used by HMM and TGMR produces very variable results. Indeed, EM is a local search procedure that starts randomly (with *k-means* initialization) and stops for example once a local maximum likelihood is reached (other stopping criterions can be defined). Depending on the initialization, a very different number of iterations may be required to reach a local optimum.[2] For reproduction, $\mathcal{M}_5$ shows that the different methods remain competitive in terms of online retrieval of data (less than 1 millisecond, and nearly linear increase for dimensions below $D = 12$).

We can conclude from this evaluation that the HMM method is competitive with respect to the other approaches. The next sections present three robot learning applications, which are aimed at showing the interesting characteristics of the proposed approach. The experiments present different learning situations where the above quantitative measures would not be be an appropriate benchmark to highlight the properties of the approach.

IV. Experiment with iCub humanoid robot

In the applications that we present next, we consider the dynamics of the movement but we do not consider the dynamics of the robot itself. In the tasks that we consider, the robot is sufficiently fast and precise to track the dynamics of the trajectory (and the inertia remains low). However, if tracking errors occur, they are intrinsically handled by the system, which acts as an autonomous system robust to perturbation. Videos of the experiments accompany the paper.

The aims of the experiment with iCub are to show that: (1) the proposed approach can be used to learn periodic motion containing crossings; and (2) the algorithm can cope with bimanual movements in joint angle space.

[2]In practice, a maximum number of iterations can be set (which was not the case in this experiment) to guarantee that the learning time remains short.

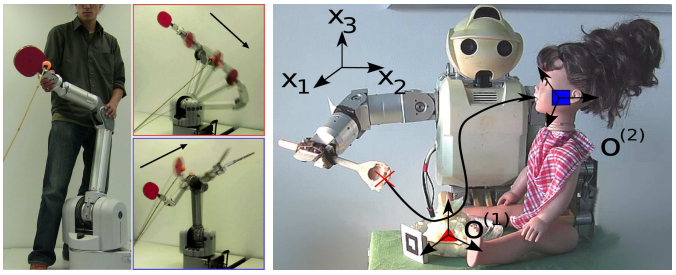

Fig. 5. *Left:* Teaching the *Barrett WAM* robotic arm to hit a ball, with reproduction of a *drive* stroke (top) and *topspin* stroke (bottom). *Right:* Teaching the HOAP-3 humanoid robot to feed a *Robota* robotic doll.

The iCub robot is used in the experiment [29]. 14 DOFs out of the 53 degrees-of-freedom (DOFs) are used to control the two arms of the robot. A set of motion sensors are used to record the user's gestures by collecting joint angle trajectories of the upper-body torso, see Fig. 4. Six *X-Sens* motion sensors are attached to the upper-arms, lower-arms, and at the back of the hands of the user, sending 14 DOFs data to the robot.

A simple rhythmic movement is demonstrated through the motion sensors and simultaneously mapped to the iCub. After having observed 3-4 periods of the movement, the robot learns a model of the cyclic motion. The motion is reproduced by the HMM approach presented in Section II, and compared to DMP. For DMP, the version of the algorithm for periodic motion is employed, where the period of the movement has been explicitly defined.

Fig. 4 presents the encoding, reproduction and evaluation results. The 14 DOFs motion contains a crossing in joint space, which is also observed in the PCA projection of the data. At a given iteration, the robot must thus move differently depending on the preceding postures and movements along the trajectory. We see that the high-dimensional periodic movement with crossing is correctly handled by DMP and HMM (8 states have been used in both cases). DMP shows the best score in terms of accuracy and smoothness. The cyclic form must however be set beforehand (discrete and periodic signals use a different representation in DMP), with an external method required to estimate the period of the motion.

In contrast to HMM and DMP, LWR and LWPR fail to reproduce the movement at the crossing point. Passing through the same point several times along the the cycle is not correctly handled here. When reaching the crossing point, these two methods retrieve an undesirable average of the different motion behaviors learned at this point. The system can also follow indefinitely only a subpart of the periodic movement. For this reason, LWR and LWPR have not been quantitatively evaluated here. Similarly, TGMR has not been evaluated as it cannot efficiently encode periodic motion due to the explicit encoding of time in the model.

V. Experiment with WAM robotic arm

This experiment shows that the framework can be used in an unsupervised learning manner. By that, we mean that several movements can be encoded in a single HMM, without specifying the number of movements, and without associating the different demonstrations with a class or label.

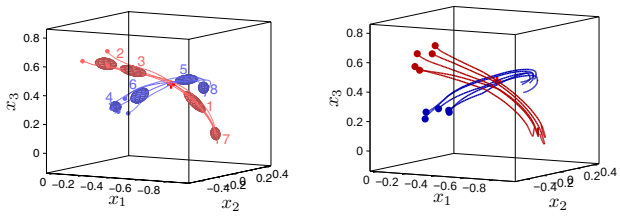

Fig. 6. Encoding and reproduction results of the table tennis experiment (in position space). *Left:* Demonstrated movements and associated Hidden Markov Model, where 8 Gaussians are used to encode the two categories of movements (the learned transitions are represented in Fig. 7). The position of the ball is depicted by a plus sign, and the initial points of the trajectories are depicted by dots. 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 in the areas where either *topspin* and *drive* strokes have been demonstrated.

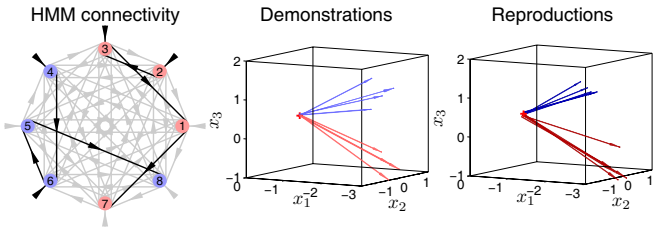

Fig. 7. *From left to right:* HMM representation of the transitions and initial state probabilities (the corresponding state output distributions are represented in Fig. 6). The states of the HMM are spatially organized around a circle for representation purposes. The possible transitions are depicted 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. Position and velocity of the racket at the time of the impact for the 8 demonstrations (*center*), and for the 10 reproduction attempts (*right*).

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. 5 *left*. One objective is to demonstrate that such movements can be transferred using the proposed approach, where the skill requires that the target be reached with a given velocity, direction and amplitude. In the experiment presented here, we extend the difficulty of the tennis task described in [13] by assuming that the robot must hit the ball with a desired velocity set by the demonstrations.

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, and 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 him to move the robot manually. Through this *kinesthetic teaching* process, the user *molds* the robot behavior by putting it through the task

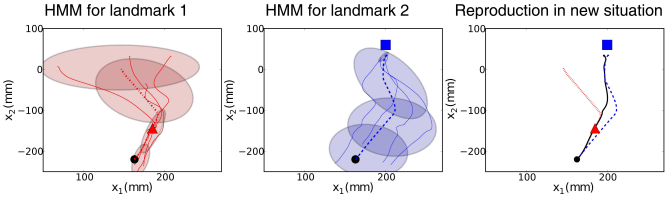

Fig. 8. Trajectories relative to the two landmarks are encoded in two HMMs of 4 states. Each Gaussian encodes position and velocity information along the task. Generated trajectories using the corresponding models are represented with dashed lines, where the dots show the initial positions. The position of the landmarks are represented with a triangle for the plate and with a square for Robota's mouth. 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 Robota's mouth).

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 color-based stereoscopic vision tracking system (Vivotek cameras with a resolution of 640x480 pixels).

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) [30]. A damped least square inverse kinematics solution with optimization in the null space of the Jacobian matrix is used to reproduce the task, see [12] for details.

Figs 5 and 6 present the encoding and play back results. We see that the HMM approach reproduces an appropriate motion in the two situations. Fig. 7 *left*, presents the states transitions learned by the HMM. We see that the model has correctly learned that two different dynamics can be achieved here, depending on the initial position of the robot. It is thus possible to encode several motion alternatives in a single model, without having to provide the number (and labels) of the movements during the demonstration phase. The different gestures are then automatically retrieved depending on the initial situation. Fig. 7 also presents the results of the strokes at the time of the impact with the ball. We see that the system correctly strikes the ball at a velocity similar to that demonstrated (in terms of both amplitude and direction).

VI. Experiment with HOAP-3 and Robota

This experiment shows that the framework can be used to learn a controller by taking simultaneously into account several constraints. Here, we consider the case where a set of movements relative to a set of landmarks must be considered for a correct reproduction of the skill (i.e., where several actions on objects are relevant for the task).

In the previous experiment, we learned trajectories in the frame of reference of a single object (the ball). This experiment with a humanoid robot extends this approach by considering

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 [31], where *HOAP-3* first brings a spoon to a plate to collect mashed potatoes and then moves it with an appropriate path towards *Robota*'s mouth, see Fig. 5. Four kinesthetic demonstrations are provided by changing the initial positions of the landmarks from one demonstration to the other.

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.[3]

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

A *Hidden Markov Model* is learned for each landmark. 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 through $\hat{x}'^{(n)} = R'^{(n)}\hat{x}^{(n)} + o'^{(n)}$ and $\hat{\dot{x}}'^{(n)} = R'^{(n)}\hat{\dot{x}}^{(n)}$.[4]

At each time step, the command defined in Eq. (2) is used to retrieve the desired velocity $\hat{\dot{x}}'$ and desired position $\hat{x}'$, where the resulting distributions $\mathcal{N}(\hat{\dot{x}}', \hat{\Sigma}'^{\dot{x}})$ and $\mathcal{N}(\hat{x}', \hat{\Sigma}'^{x})$ are respectively computed through the Gaussian products $\prod_{n=1}^{N}\mathcal{N}(\hat{\dot{x}}'^{(n)}, \hat{\Sigma}'^{\dot{x}(n)})$ and $\prod_{n=1}^{N}\mathcal{N}(\hat{x}'^{(n)}, \hat{\Sigma}'^{x(n)})$. This allows the system to combine automatically the different constraints associated with the landmarks.

Fig. 8 presents the encoding results. It shows through the patterns of the Gaussian distributions that parts of the motion are more constrained than others. With respect to landmark 1, strong consistency among the demonstrations has been observed at the beginning of the gesture (motion of the spoon into the mashed potatoes), which is reflected by the narrower form of the ellipses at this point. With respect to landmark 2 (*left* graph), strong consistency among the demonstrations has been observed at the end of the gesture (when reaching for Robota's mouth). Fig. 8, *right*, shows the reproduction results. We see that the robot automatically combines the two sets of constraints (associated with the plate and with Robota's mouth) to find a trade-off satisfying probabilistically the constraints observed during the demonstrations.

[3] *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 (a visual marker would easily be occluded by the spoon moving in the vicinity of the mouth).

[4] The associated covariances matrices are transformed through the linear transformation property of Gaussian distributions, yielding $\hat{\Sigma}'^{x(n)} = R'^{(n)}\hat{\Sigma}^{x(n)}R'^{(n)^\top}$ and $\hat{\Sigma}'^{\dot{x}(n)} = R'^{(n)}\hat{\Sigma}^{\dot{x}(n)}R'^{(n)^\top}$.

VII. Discussion

We presented an evaluation experiment based on randomly generated data and three applications highlighting different capabilities of the proposed system. The aim of the experiment presented in Section III was to perform a systematic evaluation for various dimensionalities and for models of various complexity. It, however, remains valid only for the subset of tasks that we consider here, that is, in the context where an acceleration command is recursively evaluated after having observed a set of position and velocity data. Future comparison effort is required to evaluate the different methods for a broader range of tasks.

The proposed HMM approach shares many characteristics with the DMP approach, but has some advantages for the subset of tasks that we considered in the experiments. First, it is able to encode several motion alternatives in the same model (see the table tennis experiment in Sec. V). Partial demonstrations can be provided, which is a very important characteristics for the teaching interaction (e.g. to refine one part of the movement without having to demonstrate the whole task again). Compared to DMP that must explicitly embed the cyclic or discrete form of the motion, the HMM approach allows periodic and reaching movements to be handled in a unified way (and simultaneously), without having to specify the representation beforehand (see the dance learning experiment in Sec. IV). It is thus not necessary to specify the frequency of the movement in contrast with DMP that requires the use of an external system to estimate the fundamental frequency of the system [32], [33].

DMP is robust to spatial perturbation but requires an external mechanism to handle temporal perturbations such as delay and pauses in the motion (the perturbation needs to be detected in order to re-estimate the value of the decay term). For example, if the robot needs to reproduce only one part of the motion, or if the target is moving, the decay term must be re-evaluated in consequence. Handling this type of perturbation is in contrast inherently encapsulated in the proposed model.

The proposed model allows automatic learning of the correlations between the different variables, and the use of this information for reproduction. To handle multivariate data, DMP considers the different variables as separate processes synchronized by the phase variable, whereas HMM encapsulates the complete correlation information.

The interesting properties of the proposed model however comes with a drawback that requires further investigation. In DMP, the weights are determined through a decay term, which allows the system to guarantee convergence to an attractor. In contrast, the HMM method has the disadvantage that its stability relies on the proper choice of the gains in Eq. (2). These gains must be set by estimating in advance the

perturbations that one can expect during reproduction and/or the range of novel initial positions that the system is expected to handle.

VIII. Conclusion

We presented and evaluated an approach based on Hidden Markov Model, Gaussian Mixture Regression and dynamical systems to allow robots to acquire new skills by imitation. The use of HMM allowed us to get rid of the explicit time dependency that was considered in our previous work [12], by encapsulating precedence information within the statistical representation. In the context of separated learning and reproduction processes, this novel formulation was systematically evaluated with respect to our previous approach, *Locally Weighted Regression* (LWR) [19], *Locally Weighted Projection Regression* (LWPR) [20], and *Dynamic Movement Primitives* (DMP) [13]. We finally presented applications on different kinds of robots to highlight the flexibility of the proposed approach in three different learning by imitation scenarios.

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] B. 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.

[3] J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato, "Learning from demonstration and adaptation of biped locomotion," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 79–91, 2004.

[4] M. Nicolescu and M. Mataric, "Natural methods for robot task learning: Instructive demonstrations, generalization and practice," in *Proc. Intl Joint Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, Melbourne, Australia, 2003, pp. 241–248.

[5] J. Saunders, C. Nehaniv, and K. Dautenhahn, "Teaching robots by moulding behavior and scaffolding the environment," in *Proc. ACM SIGCHI/SIGART Conf. on Human-Robot Interaction (HRI)*, Salt Lake City, Utah, USA, March 2006, pp. 118–125.

[6] M. Pardowitz, R. Zoellner, S. Knoop, and R. Dillmann, "Incremental learning of tasks from user demonstrations, past experiences and vocal comments," *IEEE Trans. on Systems, Man and Cybernetics, Part B*, vol. 37, no. 2, pp. 322–332, 2007.

[7] S. Ekvall and D. Kragic, "Learning task models from multiple human demonstrations," in *Proc. IEEE Intl Symposium on Robot and Human Interactive Communication (RO-MAN)*, Hatfield, UK, September 2006, pp. 358–363.

[8] A. Alissandrakis, C. Nehaniv, and K. Dautenhahn, "Correspondence mapping induced state and action metrics for robotic imitation," *IEEE Trans. on Systems, Man and Cybernetics, Part B*, vol. 37, no. 2, pp. 299–307, 2007.

[9] B. Argall, B. Browning, and M. Veloso, "Learning robot motion control with demonstration and advice-operators," in *IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, September 2008, pp. 399–404.

[10] 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.

[11] 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.

[12] S. Calinon and A. Billard, "Statistical learning by imitation of competing constraints in joint space and task space," *Advanced Robotics*, vol. 23, no. 15, pp. 2059–2076, 2009.

[13] 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)*, 2001, pp. 752–757.

[14] 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.

[15] D. Grimes, R. Chalodhorn, and R. Rao, "Dynamic imitation in a humanoid robot through nonparametric probabilistic inference," in *Proc. Robotics: Science and Systems (RSS)*, 2006, pp. 1–8.

[16] 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.

[17] 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. Tesauro, and J. Alspector, Eds., vol. 6. Morgan Kaufmann Publishers, Inc., 1994, pp. 120–127.

[18] 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)*, 2009, pp. 1635–1642.

[19] S. Schaal and C. Atkeson, "Constructive incremental learning from only local information," *Neural Computation*, vol. 10, no. 8, pp. 2047–2084, 1998.

[20] S. Vijayakumar, A. D'souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Computation*, vol. 17, no. 12, pp. 2602–2634, 2005.

[21] D. Nguyen-Tuong and J. Peters, "Local gaussian process regression for real-time model-based robot control," in *IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, 2008, pp. 380–385.

[22] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77:2, pp. 257–285, February 1989.

[23] 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.

[24] S. Calinon, F. D'halluin, D. Caldwell, and A. 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.

[25] M. Khansari and A. 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.

[26] S. Calinon, E. Sauser, A. Billard, and D. Caldwell, "Evaluation of a probabilistic approach to learn and reproduce gestures by imitation," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Anchorage, Alaska, USA, May 2010.

[27] D. Bullock and S. Grossberg, "Neural dynamics of planned arm movements: Emergent invariants and speed-accuracy properties during trajectory formation," *Psychological Review*, vol. 95, no. 1, pp. 49–90, 1988.

[28] N. Hogan, "Adaptive control of mechanical impedance by coactivation of antagonist muscles," *IEEE Trans. on Automatic Control*, vol. 29, no. 8, pp. 681–690, 1984.

[29] N. Tsagarakis, G. Metta, G. Sandini, D. Vernon, R. Beira, F. Becchi, L. Righetti, J. Santos-Victor, A. Ijspeert, M. Carrozza, and D. Caldwell, "iCub: The design and realization of an open humanoid platform for cognitive and neuroscience research," *Advanced Robotics*, vol. 21, no. 10, pp. 1151–1175, 2007.

[30] G. Schwarz, "Estimating the dimension of a model," *Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[31] A. Billard, "Robota: Clever toy and educational tool," *Robotics and Autonomous Systems*, vol. 42, no. 3–4, pp. 259–269, 2003.

[32] A. Gams, A. Ijspeert, S. Schaal, and J. Lenarcic, "On-line learning and modulation of periodic movements with nonlinear dynamical systems," *Autonomous Robots*, vol. 27, no. 1, pp. 3–23, 2009.

[33] J. Buchli, L. Righetti, and A. Ijspeert, "Frequency analysis with coupled nonlinear oscillators," *Physica D: Nonlinear Phenomena*, vol. 237, pp. 1705–1718, August 2008.

Sylvain Calinon is a Team Leader at the Advanced Robotics Department, Italian Institute of Technology (IIT), and a visiting researcher at the Learning Algorithms and Systems Laboratory (LASA), Swiss Federal Institute of Technology in Lausanne (EPFL). He received a PhD on robot programming by demonstration in 2007 from EPFL, which was awarded by the international Robotdalen scientific award, Asea Brown Boveri (ABB) award and EPFL-Press distinction. From 2007 to 2009, he was a postdoctoral research fellow

at LASA. His research interests cover robot learning by imitation and human-robot interaction.

Florent D'halluin is a PhD student at the Learning Algorithms and Systems laboratory at EPFL. He received a M.S, in Microeenginering and Robotics from EFPL in 2009, and he graduated from Ecole Polytechnique in France in 2008. His research interests are mainly Learning by demonstration and Machine Learning applied to robotics.

Eric L. Sauser is a Post-Doctoral Fellow with the Learning Algorithms and Systems Laboratory (LASA) at the Swiss Federal Institute of Technology, Lausanne (EPFL). He received his BSc/MSc in Computer Science from EPFL in 2001 and March 2003. He then joined the Learning Algorithms and Systems Laboratory and obtained his PhD in Computational Neuroscience in December 2007. His current research interests cover a broad range of Robotic-related topics such as Programming by Demonstration, Machine Learning, Robot Control, and Human-Robot Interaction.

Aude G. Billard is Associate Professor and head of the LASA Laboratory at the School of Engineering at the Swiss Federal Institute of Technology in Lausanne (EPFL). Prior to this, she was Research Assistant Professor at the department of Computer Sciences at the University of Southern California, where she retained an adjunct faculty position to this day. Aude Billard received a B.Sc. (1994) and M.Sc. (1995) in Physics from EPFL, with specialization in Particle Physics at the European Center for Nuclear Research (CERN), a MSc. in Knowledge-based Systems (1996) and a Ph.D. in Artificial Intelligence (1998) from the Department of Artificial Intelligence at the University of Edinburgh. Her research interests focus on machine learning tools to support robot learning through human guidance. This extends also to research on complementary topics, including machine vision and its use in human-machine interaction and computational neuroscience to develop models of learning in humans.

Darwin G. Caldwell is a Director at the Italian Institute of Technology in Genoa, Italy, and a Visiting/Honorary Professor at the Universities of Sheffield, Manchester, Bangor and Kings College, London. His research interests include innovative actuators and sensors, haptic feedback, force augmentation exoskeletons, dexterous manipulators, humanoid robotics (iCub), bipedal and quadrapedal robots (HyQ), biomimetic systems, rehabilitation robotics, telepresence and teleoperation procedures, and robotics and automation systems for the food industry. He is the author or co-author of over 200 academic papers, 8 patents and has received awards at several international conferences and events including, ICRA, IROS, ICAR, Humanoids, IAV and Virtual Concepts. He is on the editorial board of Journal for Robots, International Journal of Social Robotics and Industrial Robot and Technical Editor of the Trans. in Mechatronics.