Encoding the time and space constraints of a task in explicit-duration Hidden Markov Model

Sylvain Calinon, Antonio Pistillo and Darwin G. Caldwell

Abstract— **We study the use of different weighting mechanisms in robot learning to represent a movement as a combination of linear systems. Kinesthetic teaching is used to acquire a skill from demonstrations which is then reproduced by the robot. The behaviors of the systems are analyzed when the robot faces perturbation introduced by the user physically interacting with the robot to momentarily stop the task. We propose the use of a Hidden Semi-Markov Model (HSMM) representation to encapsulate duration and position information in a robust manner with parameterization on the involvement of time and space constraints. The approach is tested in simulation and in two robot experiments, where a 7 DOFs manipulator is taught to play a melody by pressing three big keys and to pull a model train on its track.**

I. INTRODUCTION

One building block common to various approaches in robot learning is the representation of tasks and nonlinear movements. We take the perspective that a superposition of linear subsystems can represent the realm of the continuous world, and that such representation can be manipulated/visualized by users and help at generalizing the task to new situations. Various representations of movement primitives have been adopted, with the common perspective that the combination of linear subsystems $\boldsymbol{f}_i(\boldsymbol{x})$ can lead to complex behaviors

$$\dot{\boldsymbol{x}} = \sum_{i=1}^{K} h_i(\boldsymbol{x},t)\; \boldsymbol{f}_i(\boldsymbol{x}) = \sum_{i=1}^{K} h_i(\boldsymbol{x},t)\; \Big(\boldsymbol{A}_i\boldsymbol{x} + \boldsymbol{b}_i\Big). \quad (1)$$

Examples of models that can be formulated in this way are the *Stable Estimator of Dynamical Systems* (SEDS) [1], the *Dynamic Movement Primitives* (DMP) [2], [3] (see [4] and [5] for explanations of the reformulation), the *Takagi-Sugeno Model* (TSM) [6], or the biologically-inspired computational models combining irrotational and solenoidal vector fields [7]. These methods differ in the representation of $\boldsymbol{x}$,[1] in the way $\boldsymbol{A}_i$ and $\boldsymbol{b}_i$ are estimated and constrained, and in the mechanism that combines the different subsystems through scalar weights h_i.

We concentrate in this paper on this last issue, which has often been put aside but which is a crucial aspect that modifies the behavior of a system under strong perturbation. We first review different weighting mechanisms to switch between movement primitives along the task, by emphasizing that these mechanisms are often interchangeable across the different models, see e.g. [8], [9]. Resulting from this analysis, we then introduce a new weighting scheme that encompasses spatial, temporal and sequential perspectives within a unified notation based on *Hidden Semi-Markov Model* (HSMM) [10].

The core *Hidden Markov Model* (HMM) structure offers a multitude of variants covering various robot learning issues in real-world environment. In robot learning by imitation, it presents relevant characteristics to handle spatial and temporal variabilities across multiple demonstrations, see e.g. [9], [11]–[15]. This structure is used here to study the different forms of recovery that the robot can use when the system is faced with strong perturbations such as stopping the robot in the course of the task, or pushing the robot away from its reproduced trajectory [16].

II. COMBINATION OF MOVEMENT PRIMITIVES

The problem of combining efficiently movement primitives has been addressed with perspectives and formulations in biology, robotics and system control. In its most simple form, the superposition of movement primitives relates to control approaches based on piecewise linear systems (see e.g. [17]) defined by step-function switching rules $h_i^{\text{PLS}} = B_i(\boldsymbol{x},t)$ with $B_i = \{0,1\}$. Approaches in Fuzzy Logic such as the *Takagi-Sugeno Model* (TSM) [6] extends the weighting mechanism with linear switching rules $h_i^{\text{TSM}} = A_i t + b_i$, paralleling the subjective description of tasks in human language.

Computational framework have been explored in biology to superpose independent elementary vector fields to study movement learning and control in both biological and artificial systems [7]. The aim of such models is to show that the task of generating coordinated movements in a multi-articular system can be carried out by combining vector fields generated by independent modules of control. One of this pioneer work considers a linear combination of irrotational and solenoidal fields [7], where several sets of weights can be considered to reproduce different behaviors.

A. Weights based on Gaussian distributions in time

Several weighting mechanisms do not use time directly but create instead an implicit time dependency on the system [4], [8].[2] For example, the weighting mechanism in *Dynamic Movement Primitives* (DMP) [2], [3] is defined by a decay

The authors are with the Department of Advanced Robotics, Istituto Italiano di Tecnologia (IIT), 16163 Genova, Italy. {`sylvain.calinon`, `antonio.pistillo`, `darwin.caldwell`}`@iit.it`.

[1]For example, a second order system can be defined for DMP by considering the state $\boldsymbol{x} = \binom{\boldsymbol{\theta}}{\dot{\boldsymbol{\theta}}}$ in which $\boldsymbol{\theta}$ and $\dot{\boldsymbol{\theta}}$ are angular positions and velocities, see [5] for details.

[2]The aim of such reparameterization is to include some modulation properties.

term evolving through a dynamical system $\dot{s} = -\omega s$, initialized with $s = 1$ and converging to zero Even though the weighting mechanism is originally defined on s, a common practice is to define weighting kernels that equivalently have equal spacing in time, see [3], [8].

A similar time-based mechanism can be defined as Gaussian distributions (or equivalently radial basis functions) with predetermined variance and centers equally distributed in time

$$h_i^{\text{TIME}} = \frac{\alpha_i^{\text{TIME}}}{\sum_{k=1}^K \alpha_k^{\text{TIME}}}, \quad \text{with } \alpha_i^{\text{TIME}} = \mathcal{N}(t;\ \mu_i^{\mathcal{T}}, \Sigma_i^{\mathcal{T}}). \tag{2}$$

B. Gaussian Mixture Model and Hidden Markov Model

Approaches based on *Gaussian Mixture Model* (GMM) and *Gaussian Mixture Regression* (GMR) [1], [9] can provide autonomous systems based on time-invariant processes, with

$$h_i^{\text{GMM}} = \frac{\alpha_i^{\text{GMM}}}{\sum_{k=1}^K \alpha_k^{\text{GMM}}}, \quad \text{with } \alpha_i^{\text{GMM}} = \mathcal{N}(\boldsymbol{x};\ \boldsymbol{\mu}_i^{\mathcal{X}}, \boldsymbol{\Sigma}_i^{\mathcal{X}}). \tag{3}$$

The HMM representation presents an intermediary approach where the *forward* variable is used as a weight encapsulating spatial and sequential information [18]. Here, the output distribution of each HMM state is represented by a Gaussian locally encoding variation and correlation information. The parameters of the HMM $\{\pi_i, a_{i,j}, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}_{i=1}^K$ are learned by *Expectation-Maximization* (EM), where π_i is the initial probability of being in state i and $a_{i,j}$ is the transitional probability from state i to state j. $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ represent the center and the covariance matrix of the i-th Gaussian (one Gaussian per HMM state is considered here). By using this representation, the *forward* variable is used in [9] to combine the movement primitives with weights recursively computed as

$$\begin{aligned} h_{i,n}^{\text{HMM}} &= \frac{\alpha_{i,n}^{\text{HMM}}}{\sum_{k=1}^K \alpha_{k,n}^{\text{HMM}}}, \quad \text{with} \\ \alpha_{i,n}^{\text{HMM}} &= \Big(\sum_{j=1}^K \alpha_{j,n-1}^{\text{HMM}}\ a_{j,i}\Big)\mathcal{N}(\boldsymbol{x}_n;\ \boldsymbol{\mu}_i^{\mathcal{X}}, \boldsymbol{\Sigma}_i^{\mathcal{X}}), \end{aligned} \tag{4}$$

and initialization given by $\alpha_{i,1}^{\text{HMM}} = \pi_i\ \mathcal{N}(\boldsymbol{x}_1;\ \boldsymbol{\mu}_i^{\mathcal{X}}, \boldsymbol{\Sigma}_i^{\mathcal{X}})$.

If one wishes to use only transition probability information (discarding spatial information) to estimate the weights, the forward variable can be initialized with $\alpha_{i,1}^{\text{HMM}} = \pi_i$, and recursively computed with $\alpha_{i,n}^{\text{HMM}} = \sum_{j=1}^K \alpha_{j,n-1}^{\text{HMM}}\ a_{j,i}$. The abbreviation tHMM will be used in this case.

Even if standard HMM encapsulates spatial and sequential information, the *forward* variable in (4) exponentially biases position information over transition information. Indeed, temporal information is only poorly represented by the model through homogeneous transition probabilities. Several directions have been proposed to provide a better equity between duration and space information.

Lee and Ott proposed in [14] to combine HMM with *Gaussian Mixture Regression* (GMR) to counter the limitations of the simple modeling of transition information in standard HMM to retrieve smooth trajectories. In their work, the efficiency of HMM for online incremental recognition is combined with the efficiency of GMR for smooth trajectory retrieval based on explicit time modeling.

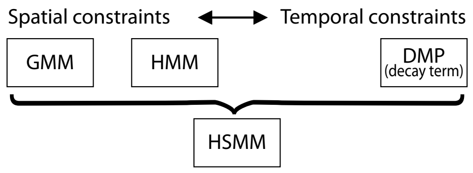

Fig. 1. The *Hidden Semi-Markov Model* (HSMM) representation allows us to parameterize the privileges for time and space information. At one utmost limit, the system produces transitions across a set of primitives with predefined duration and blending (such as in an explicit time-dependent model). At the other utmost limit, the system produces weights based solely on the current position (state) of the system, emulating an autonomous system with GMM weighting mechanism. The probabilistic modeling of state duration in HSMM allows the robot to cover a wider range of behaviors for reproduction under strong perturbation.

Kim *at al* proposed in [19] to cope with the limitations of weights based on GMM when one wishes to reproduce motion with new timing constraints. The total duration of the remaining movement is estimated by internally simulating reproduction until convergence. This information is then used to define a scalar gain in the form of a dynamical system that modulates the velocity computed in (1).

C. Hidden semi-Markov model

We explore here the use of *Hidden Semi-Markov Model* (HSMM) [10] to switch across dynamical systems by taking into account temporal and/or spatial information. To illustrate the approach, we will consider a physical human-robot interaction example in which the user temporarily holds the end-effector of the robot while it executes a skill. The robot is gravity compensated and controlled with low gains to provide a safe bidirectional interaction. In this context, the desired recovery can be drastically different depending on the executed skill. For example, the robot should continue its movement without changing the path for a writing task (spatial information prevails). For a drumming task, the robot should recover from the delay involved by the perturbation by skipping unimportant parts of the movement to reproduce an adequate rhythmic behavior. When faced with perturbation, most of the tasks require to set a compromise between space and time constraints (with possible modification of these constraints during the task).

Retrieving duration information based on the fixed set of self-transition probabilities of conventional HMM does not provide a good model of state duration. It is however possible to modify the structure of the HMM to replace the self-transition probability $a_{i,i}$ of each state i with a parametric model $p_i(d)$ of the state duration $d \in \{1, 2, \ldots, d^{\max}\}$, where $d^{\max}$ determines a maximum limit for the number of iterations that the system can stay in a state.[3]

[3]Note that by setting $p_i(d)$ to an exponential density, a standard HMM is obtained. In practice, $d^{\max}$ can be set by dividing the length of the demonstration by the number of states in the model, and defining $d^{\max}$ as 2-3 times this value to guarantee that the duration probability distribution is well defined even if EM converges to poor local optima. Defining $d^{\max}$ as the length of the trajectory is also possible but could be computationally expensive for long movement.

There are several duration models in the HMM literature sharing this core idea, see e.g. [10], [18]. We have selected the HSMM because it provides a structure that can be analyzed, visualized and modified in the most intuitive way, which is a desirable feature in our application, see Fig. 1.

The rescaled *forward* variable of HSMM is defined as

$$h^{\text{HSMM}}_{i,n} = \frac{\alpha^{\text{HSMM}}_{i,n}}{\sum_{k=1}^{K}\alpha^{\text{HSMM}}_{k,n}}, \quad \text{with} \tag{5}$$

$$\alpha^{\text{HSMM}}_{i,n} = \sum_{j=1}^{K}\sum_{d=1}^{\min(d^{\max},n-1)} \alpha^{\text{HSMM}}_{j,n-d}\, a_{j,i}\, p_i(d) \prod_{s=n-d+1}^{n}\mathcal{N}(\boldsymbol{x}_s; \boldsymbol{\mu}^{\mathcal{X}}_i, \boldsymbol{\Sigma}^{\mathcal{X}}_i),$$

and initialization given by $\alpha^{\text{HSMM}}_{i,1} = \pi_i\, \mathcal{N}(\boldsymbol{x}_1;\ \boldsymbol{\mu}^{\mathcal{X}}_i, \boldsymbol{\Sigma}^{\mathcal{X}}_i)$, see [10] for details.

If one wishes to use only state duration and transition information to estimate the weights (thus discarding spatial information), the forward variable is initialized with $\alpha^{\text{tHSMM}}_{i,1} = \pi_i$, and recursively computed with $\alpha^{\text{tHSMM}}_{i,n} = \sum_{j=1}^{K}\sum_{d=1}^{\min(d^{\max},n-1)} \alpha^{\text{tHSMM}}_{j,n-d}\ a_{j,i}\ p_i(d)$. The abbreviation tHSMM will be used in this case.

The *forward-backward* (FB) procedure is the key process for HMM parameters learning and likelihood estimation. For standard HMM, the algorithm stores the state probabilities for one time step, while for HSMM, it requires a state probability history determined by $d^{\max}$ iteration steps, see (4) and (5). The complexity of computing the *forward* variable in standard HMM is $\mathcal{O}\left((D^2+2)KN\right)$ for a left-right model, with single Gaussian per state and full covariance (D is the dimension of each datapoint, N the number of datapoints, and K the number of HMM states).

With [10], the complexity of computing the HSMM *forward* variable is $\mathcal{O}\left((D^2+1+d^{\max})KN\right)$. Practically, if movements are recorded at a higher rate than the relevant dynamics to acquire, the training dataset can be downsampled to reduce N and $d^{\max}$. Even if the HSMM forward variable computation is slower than HMM, it can be used in practice for online robot applications (practical computation time will be provided in Sec. III-A). In our paper, the proposed algorithms are adapted to parametric state duration $p_i(d) = \mathcal{N}(d\Delta t;\ \mu^{\mathcal{D}}_i, \Sigma^{\mathcal{D}}_i)$ and parametric output distributions represented by Gaussians $\mathcal{N}(\boldsymbol{\mu}^{\mathcal{X}}_i, \boldsymbol{\Sigma}^{\mathcal{X}}_i)$.[4]

D. Generalized formulation

The α_i term in (2), (3) and (5) can be reformulated as[5]

$$\alpha^{\text{G}}_{i,n} = \Big[\alpha'_{i,n}\Big]^{w_1}\Big[\sum_{j=1}^{K}\sum_{d=1}^{\min(d^{\max},n-1)}\alpha''_{i,n}\ [\alpha'''_{i,j,n}]^{w_1 w_2}\Big]^{w_2}, \text{with} \tag{6}$$

$$\alpha'_{i,n} = \mathcal{N}(\boldsymbol{x}_n;\ \boldsymbol{\mu}^{\mathcal{X}}_i, \boldsymbol{\Sigma}^{\mathcal{X}}_i), \quad \alpha''_{i,j,n} = \alpha^{\text{G}}_{j,n-d}\, a_{j,i}\, p_i(d),$$

$$\text{and} \quad \alpha'''_{i,n} = \prod_{s=n-d+1}^{n-1}\mathcal{N}(\boldsymbol{x}_s;\ \boldsymbol{\mu}^{\mathcal{X}}_i, \boldsymbol{\Sigma}^{\mathcal{X}}_i).$$

[4]The interested readers can refer to [10] for details concerning the computation of forward-backward variables and Gaussian distribution reestimation/evaluation processes.

[5]This can be seen by factorization of the last product term in (5) for the time step t.

In the above equation, the weighting schemes corresponding to a GMM, HSMM and tHSMM representation are retrieved respectively with $\{w_1=1, w_2=0\}$, $\{w_1=1, w_2=1\}$ and $\{w_1=0, w_2=1\}$.

III. SIMULATION EXPERIMENTS

An S-shaped trajectory in a two-dimensional space is used to highlight the properties of the approach and contrast it to other weighting schemes. Each model uses 6 states and shares the same output distributions (trained by EM). Matrices $\boldsymbol{A}_i$ and vectors $\boldsymbol{b}_i$ are also shared by the different methods. These parameters have been estimated as an optimization under constraint problem[6] in which one searches for $\boldsymbol{A}_i$ and $\boldsymbol{b}_i$ values that match the demonstrated data, under the constraint that the symmetric part of $\boldsymbol{A}_i$ (namely $\frac{1}{2}(\boldsymbol{A}_i+\boldsymbol{A}_i^{\top})$) has eigenvalues with strictly negative real part (one of the optimization criterion employed in SEDS [1], see also the contracting properties of a linear time-varying system in contraction theory [20]).

A. Simulation experiments results

Fig. 2 presents the simulation results. The *top-left* plots show the result of the HSMM encoding, representing both state duration (*left*) and output distribution (*right*) with Gaussian distributions. The graph in the center illustrates the difference of connectivity between an HMM (*top*) and HSMM (*bottom*). The state duration information $p_i(d)$ shows that the states in blue are quickly visited compared to the state in orange that shows a longer duration and larger variation.

The *top-right* timeline graphs show the trajectories reproduced by HMM, HSMM and tHSMM with perturbation, respectively represented with red, blue and green lines. The demonstration is represented with black lines. We see that after perturbation, the tHSMM trajectory skips some parts of the motion to recover from the perturbation. For HMM, the shape of the trajectory is only slightly distorted, i.e., both temporal and spatial aspects are taken into consideration, but the spatial component remains dominant. HSMM shows a similar behavior, but provides a better balance between temporal and spatial constraints.

The other plots in Fig. 2 show the reproduction behaviors in normal and perturbed conditions for each model. For each model, the first timeline graph shows the changes of weights h_i for a reproduction without external perturbation (with $T=200$ iterations as in the demonstration). The second timeline graph shows the results of an hard-coded perturbation that simulates the user holding the arm of the robot during a few iterations. In the range of iteration steps $t=[95,160]$, the system is forced to stay at the same position. To show the effect of the perturbation, the number of iterations has been increased to $T=260$. Note that the system is not aware of the type, occurrence and duration of the perturbation.

For each model, the square graph on the right shows the reproduced trajectories in 2D space, where the color of each

[6]We used an interior point solver for non-linear programming initialized by weighted least-squares.

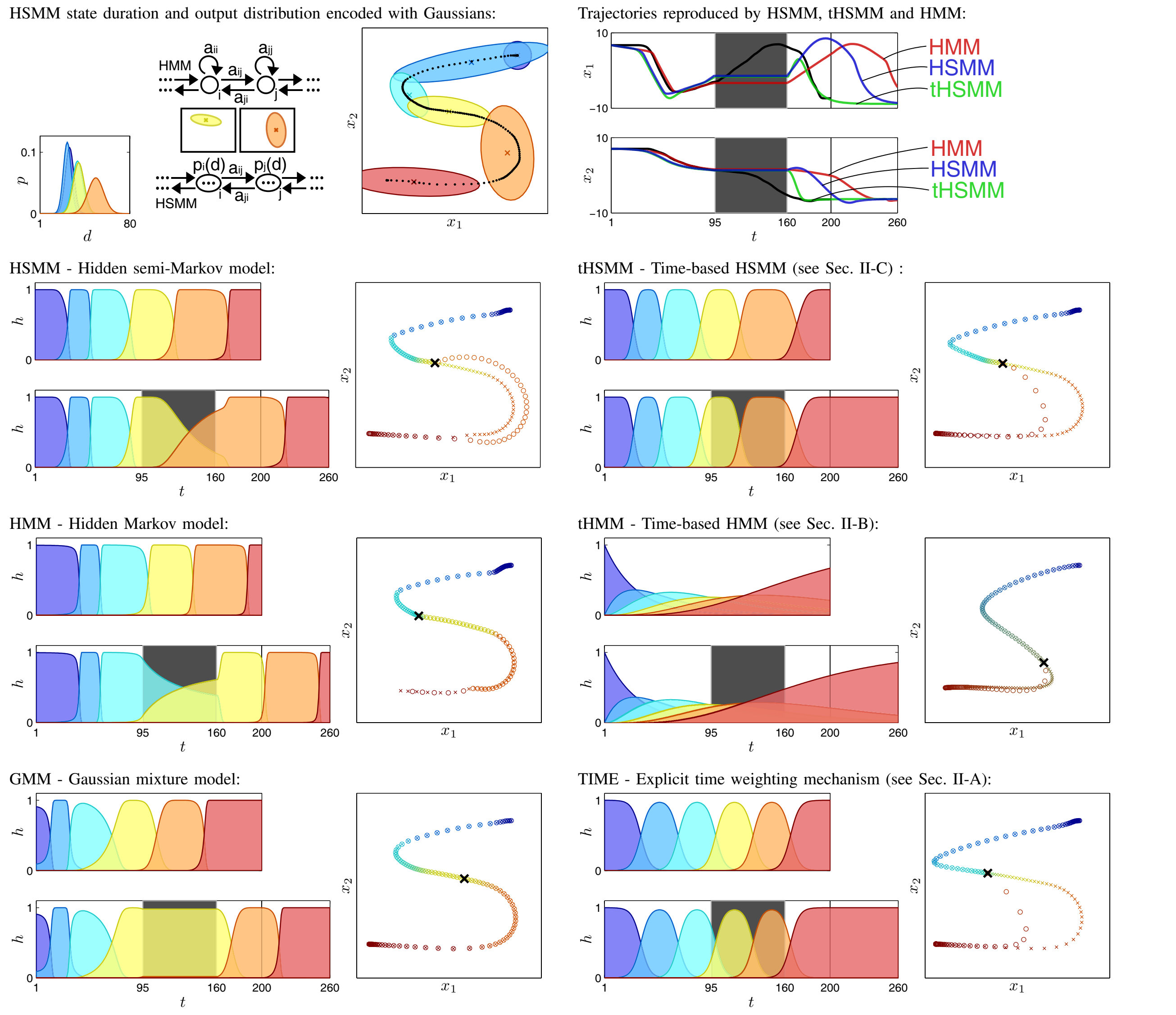

Fig. 2. Learned models and reproduction attempts. For each model, the first and second timeline graphs show respectively the normal reproduction and the reproduction with perturbation.

datapoint is blended according to the probability of belonging to the state of corresponding color. The trajectories with crosses and circles correspond respectively to unperturbed and perturbed reproductions. The thick black cross represents the position where the perturbation occurs.

Without perturbation, we see that all models reproduce a similar shape, with tHMM retrieving the worst match with strong smoothing of the original trajectory. This confirmed our expectation, since only transition probabilities are used at each iteration in tHMM to model the duration of each state, which rapidly accumulates uncertainty after a couple of iterations, as explained in Sec. II-C. TIME and tHSMM share similar characteristics by representing state transition information through an explicit encoding of time (for TIME) and state duration (for tHSMM). The weighting mechanism in TIME is characterized by a constant duration for each state as well as a constant uncertainty for each transition between two states, which is reflected by the regular shape of h_i in the timeline graphs. In contrast, tHSMM allows to set flexibly different duration for each state and different uncertainty on the transition time, while keeping a smooth blending of the different subsystems. This allows the system to cope with trajectories presenting different levels of granularity or variable complexity along the movement.[7]

Reproductions with perturbation show different behaviors. On the one hand, the systems focusing on temporal information (tHSMM, tHMM and TIME) try to recover from the perturbation by skipping some states to recover from the induced time delay. By comparing HSMM and tHSMM in the square graphs, we see that after the perturbation, tHSMM takes a shortcut to finish the trajectory. On the other

[7]Note that by setting the same center and the same variance for the duration of each state, the weighting mechanism in tHSMM can simulate the weighting scheme in TIME by modeling probabilistically the duration of a state instead of an explicit time variable.

hand, the GMM weighting mechanism (based only on spatial information) continues the iteration after the perturbation as if it did not occur (see the freezed h_i values for GMM during the perturbation). The models taking into account both temporal and spatial information show an intermediary behavior. We see that for HSMM, at the beginning of the perturbation, the weight corresponding to the yellow state first keeps the same value for a few iterations and then progressively decreases until the next state in orange is triggered.

The computation of each iteration step in (5) (HSMM) and (4) (HMM) respectively takes 0.4ms and 0.3ms in our implementation with a standard laptop running Matlab, which is sufficiently low for on-line computation in our application (the control loop is 2ms).

IV. REAL-WORLD ROBOT EXPERIMENTS

The experiment is implemented with a *Barrett WAM* 7 DOFs robotic arm, controlled by inverse dynamics solved with a recursive Newton-Euler formulation. A gravity compensation force is added to each link, and tracking of a desired path in Cartesian space is achieved by a force command $\boldsymbol{F} = m\ddot{\boldsymbol{x}}$, where m is a virtual mass and $\ddot{\boldsymbol{x}}$ is a desired acceleration command estimated by numerical differentiation. This makes the robot actively compliant and allows the user to stop the robot by holding its end-effector.

Two tasks are evaluated. The first consists of sequentially pressing three large buttons to play a melody.[8] The second task consists of guiding a model train on its track to pass over a bridge. Two demonstrations are provided for each task through kinesthetic teaching, by physically moving the robot's arm through the task while the robot is gravity compensated. Each demonstration collects 3D Cartesian coordinates of the robot's end-effector during the movement. 11 and 8 states are respectively used for the first and second tasks to encode the movement. [9] The general weighting mechanism in (6) is used for the two tasks, by setting parameters $\{w_1 = 0, w_2 = 1\}$ for the melody playing task (corresponding to the tHSMM formulation), and $\{w_1 = 1, w_2 = 0\}$ for the model railway task (corresponding to the GMM formulation). The learned model is used to reproduce the task, during which the user briefly holds the robot's end-effector to introduce perturbation.

A. Robot experiments results

Fig. 3 presents the results of the experiments. For each task, the columns from left to right show the setup of the experiment, the demonstrations and learned Gaussian distributions, the reproduction behaviors under perturbation, and the blending of the motion primitives according to the weights h_i (the perturbation is represented by the superimposed gray area). The two weighting mechanisms produce different behaviors, by continuing the progress of the weights in the melody playing task, and by "freezing" the current weights in the model railway task. In the melody playing task, the path followed by the robot takes a shortcut to recover from the time delay induced by the perturbation. The slight tracking errors observed in the graphs are due to the low gains used during reproduction to permit physical interaction with the robot. A video of the experiments accompanies the submission (video and sourcecode available on `http://programming-by-demonstration.org`).

We also reproduced the two tasks by interchanging the weighting mechanisms ($\{w_1 = 0, w_2 = 1\}$ for the model railway task, and $\{w_1 = 1, w_2 = 0\}$ for the melody playing task), in which the user applied again a similar perturbation during reproduction (similar location and duration). For the first task, the train quickly goes off its track after perturbation (see accompanying video). For the melody playing task, we analyzed the durations of subparts of movement based on the recorded videos.

To reduce the unpredictable effect of the collision with the object and the inconsistency of the sound emitting device among multiple reproduction trials, we took an average duration between two sound emissions and motion between two hits (when the robot leaves the white button and reaches the red button), which gives 2.5 ± 0.5s during reproduction for both models without perturbation. For a perturbation of 1.1 ± 0.1s, the "correct" and "wrong" models respectively produce durations of 2.5±0.5s (similar timing) and 4.0±0.5s (introducing an undesired delay). These first two experiments based on an HSMM weighting scheme confirms our expectation that there is a non-negligible difference among the different weighting mechanisms. A thorough analysis with different types of perturbation will be part of future work.

V. CONCLUSION AND FUTURE WORK

We have demonstrated that different reproduction behaviors can be reproduced by using an HSMM representation of the movement with a generalized formulation encapsulating different weighting strategies. We showed through simulated experiments and through two real-world human-robot interaction examples that different reproduction behaviors can be obtained, which is particularly noticeable when strong sources of perturbation are considered during the reproduction. We focused on the scenario where the user can physically interact with the robot to provide demonstration and to momentary halt (or slow down) the robot during reproduction. The aim of such perturbation is to provide an illustrative example to intuitively interpret the underlying mechanism of the different approaches. The proposed system is however not limited to this specific type of perturbation, since the robot does not need to explicitly detect the perturbation through an external mechanism.

We plan in future work to exploit the HSMM properties to automatically extract the most relevant weighting mechanism (namely, the parameters w_1 and w_2 in (6)) from a series of demonstrations, depending on the task and expected perturbation. More complex bi-direction teaching interaction

[8]An interface has been developed that connects the buttons to the PC controlling the robot to emit distinctive sounds when being activated.

[9]The number of states has been estimated empirically in these experiments, but automatic model selection such as Bayesian Information Criterion (BIC) could also be used.

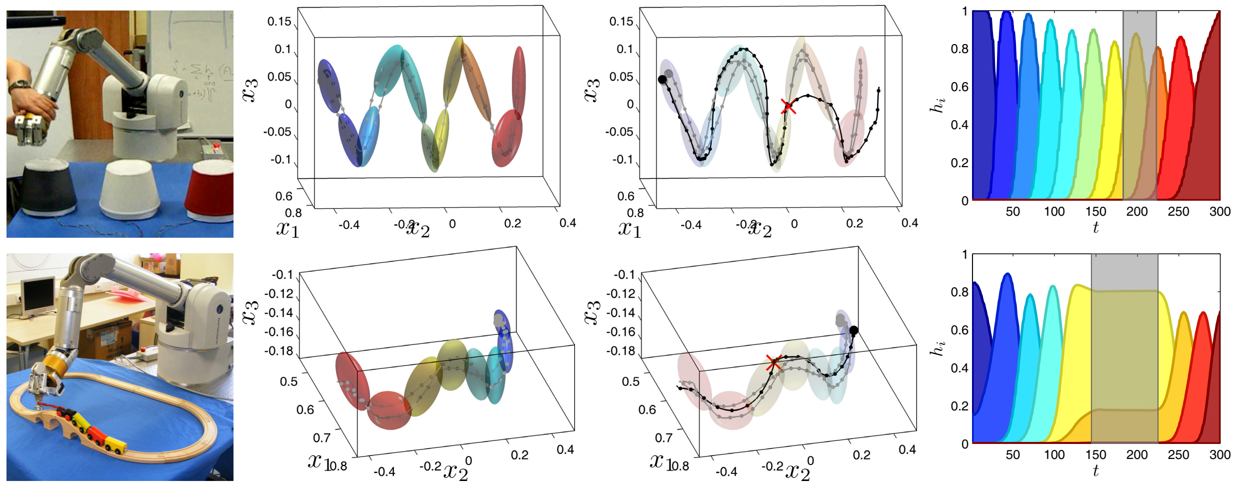

Fig. 3. *Top:* Melody playing task in which temporal constraints are important to reproduce a desired melody under perturbation. *Bottom:* Model railway task in which spatial constraints prevail when perturbation occurs. Namely, the robot needs to follow a specific path to keep the train on its track. The starting position is depicted by a large point, and the position of the perturbation is depicted by a red cross.

will be considered in which the demonstration phase is not separated with the reproduction phase. This is a particularly challenging task because on the one hand, it is not possible to automatically extract information on the weighting mechanism if noise-free demonstrations of the skill are provided (such as in scaffolded environment). On the other hand, it is not possible to robustly extract the characteristics of the task from few demonstrations if several sources of perturbations simultaneously occur. It is thus required to explore bidirectional teaching interaction in which both the user and the robot can set, test and evaluate reproductions in new situations to determine in which manner the acquired skill can be generalized while handling perturbation. We also plan to study if the binary parameters w_1 and w_2 in (6) could be defined as continuous variables to reproduce skills with time and space constraints whose levels of importance vary during the task.

References

[1] S. M. Khansari-Zadeh and A. Billard, “Imitation learning of globally stable non-linear point-to-point robot motions using nonlinear programming,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, 2010, pp. 2676–2683.

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

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

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

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

[6] T. Takagi and M. Sugeno, “Fuzzy identification of systems and its applications to modeling and control,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 15, no. 1, pp. 116–132, 1985.

[7] F. A. Mussa-Ivaldi, “From basis functions to basis fields: vector field approximation from sparse data,” *Biological Cybernetics*, vol. 67, no. 6, pp. 479–489, 1992.

[8] S. Calinon, I. Sardellitti, and D. G. Caldwell, “Learning-based control strategy for safe human-robot interaction exploiting task and robot redundancies,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, October 2010, pp. 249–254.

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

[10] S.-Z. Yu and H. Kobayashi, “Practical implementation of an efficient forwardbackward algorithm for an explicit-duration hidden Markov model,” *IEEE Trans. on Signal Processing*, vol. 54, no. 5, pp. 1947–1951, 2006.

[11] T. Asfour, F. Gyarfas, P. Azad, and R. Dillmann, “Imitation learning of dual-arm manipulation tasks in humanoid robots,” in *Proc. IEEE-RAS Intl Conf. on Humanoid Robots (Humanoids)*, Genova, Italy, December 2006, pp. 40–47.

[12] D. Kulic, W. Takano, and Y. Nakamura, “Representability of human motions by factorial hidden Markov models,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, October 2007, pp. 2388–2393.

[13] J. Butterfield, S. Osentoski, G. Jay, and O. Jenkins, “Learning from demonstration using a multi-valued function regressor for time-series data,” in *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*, Nashville, TN, USA, December 2010, pp. 328–333.

[14] D. Lee and C. Ott, “Incremental motion primitive learning by physical coaching using impedance control,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Taipei, Taiwan, October 2010, pp. 4133–4140.

[15] V. Krueger, D. L. Herzog, S. Baby, A. Ude, and D. Kragic, “Learning actions from observations: Primitive-based modeling and grammar,” *IEEE Robotics and Automation Magazine*, vol. 17, no. 2, pp. 30–43, 2010.

[16] A. Pistillo, S. Calinon, and D. G. Caldwell, “Bilateral physical interaction with a robot manipulator through a weighted combination of flow fields,” in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, San Francisco, CA, USA, September 2011.

[17] M. T. Rosenstein, A. G. Barto, and R. E. A. Van Emmerik, “Learning at the level of synergies for a robot weightlifter,” *Robotics and Autonomous Systems*, vol. 54, no. 8, pp. 706–717, 2006.

[18] B. Juang, L. Rabiner, S. Levinson, and M. Sondhi, “Recent developments in the application of hidden Markov models to speaker-independent isolated word recognition,” in *IEEE Intl Conf. on Acoustics, Speech, and Signal Processing*, vol. 10, April 1985, pp. 9–12.

[19] S. Kim, E. Gribovskaya, and A. Billard, “Learning motion dynamics to catch a moving object,” in *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*, Nashville, TN, USA, December 2010, pp. 106–111.

[20] W. Lohmiller and J. J. E. Slotine, “On contraction analysis for nonlinear systems,” *Automatica*, vol. 34, no. 6, pp. 683–696, 1998.