BM: An Iterative Algorithm to Learn Stable Non-Linear Dynamical Systems with Gaussian Mixture Models

S. Mohammad Khansari-Zadeh and Aude Billard
Ecole Polytechnique Federale de Lausanne, LASA Laboratory
{mohammad.khansari,Aude.Billard}@epfl.ch

***Abstract*— We model the dynamics of non-linear point-to-point robot motions as a time-independent system described by an autonomous dynamical system (DS). We propose an iterative algorithm to estimate the form of the DS through a mixture of Gaussian distributions. We prove that the resulting model is asymptotically *stable* at the target. We validate the accuracy of the model on a library of 2D human motions and to learn a control policy through human demonstrations for two multi-degrees of freedom robots. We show the real-time adaptation to perturbations of the learned model when controlling the two kinematically-driven robots.**

I. Introduction

We consider robot tasks that can be decomposed into sequences of point-to-point motions, i.e. movements in space stopping at a given target [1]. Modeling point-to-point motions thus provides basic components, so-called motion primitives, for robot control [1], [2]. These motions primitives can be seen as a basis, from which multiple desired robot tasks can be formed. As an example, consider the standard "pick-and-place" task, which can be decomposed as follows: First reach to the item, then after grasping move to the target location, and finally return home after release. Programming by Demonstrations (PbD) can be used to learn such motion primitives from a few demonstrations[1] of the motions performed by a trained agent (human or robot) [1], [3]–[5]. In this paper, we focus on learning point-to-point motions using PbD. When learning point-to-point motions several desiderata should be taken into account: the system should be *robust* to *temporal* and *spatial perturbation*[2], it should *autonomously adapt* its parameter to suit change in the motion dynamics and in the complexity of the path, and it should *generalize* to be applicable to contexts not seen during training.

Dynamical systems (DS) have been recently advocated as a powerful means of modeling robot motions [5]–[7]. In this paper, we take such an approach and represent a demonstrated motion as an autonomous (time independent) non-linear first order Ordinary Differential Equation (ODE). Autonomous ODE models have the advantage of being *inherently robust* with respect to *temporal perturbations*. Furthermore, these models are *flexible* in terms of generalizing a motion to parts of space not seen before, and can *immediately adapt* to *spatial perturbations*. We first show that though existing regression techniques can be used to estimate the underlying ODE, they fail to handle the classical problem of ODE functions that has been posed repeatedly in the field of dynamics, i.e. stability. Therefore, we propose a learning procedure, called Binary Merging (BM), that tackles the problem of estimating (identifying) an unknown non-linear dynamical system from a few demonstrations while ensuring its global stability.

In comparison, classical approaches such as spline-based methods [8], [9] or a range of alternative techniques using non-linear regression techniques that have been proposed over the years [2], [3] suffer from explicit time-dependency, which makes them sensitive to both temporal and spatial perturbations. To compensate, one needs a heuristic to re-index a new trajectory in time, while optimizing a given cost function that measures how well the new trajectory follows the desired one. Finding a good heuristic is highly task-dependent and a non trivial problem, and thus it becomes particularly no intuitive when considering high-dimensional state spaces [5].

This paper is structured as follows. In Section II, we first formalize the problem as a stochastic system composed of a mixture of Gaussians. In Section III, we briefly review the shortcomings of existing methods. In Section IV we develop conditions for ensuring stability of stochastic systems, and in Section V we describe an iterative method to build a Gaussian Mixture that satisfies these conditions. In Section VI, we present the experimental validation of the method, and finally we devote Section VII to summary and discussions. Further materials on BM are available at:

http://lasa.epfl.ch/sourcecode/

II. Problem Statement

Definition 1 *Let the set of N demonstrations $\{\xi^{t,n}, \dot{\xi}^{t,n}\}_{n=1,t=1}^{N,T^n}$ be instances of a global motion model governed by a first order autonomous ordinary differential equation (ODE):*

$$\dot{\xi}^{t,n} = f(\xi^{t,n}) \tag{1}$$

where $\xi^{t,n} \in \Re^d$, its time-derivative $\dot{\xi}^{t,n} \in \Re^d$ are vectors describing completely the robot's motion[3], and $f : \Re^d \to \Re^d$

[1] A demonstration is composed of a set of time sequenced points in either cartesian or joint space that forms a motion.

[2] Temporal perturbation causes the robot execution to be delayed (e.g. when slowed down because of friction in the gears) while spatial perturbations causes the robot to depart from its original trajectory (e.g. when slipping or hitting an object).

[3] E.g. ξ could be a robot's joint angles or the position of an arm's end-effector in the Cartesian space, and $\dot{\xi}$ the first order derivative of the latter.

is a non-linear continuous and continuously differentiable function with a single equilibrium point $\dot{\xi}^* = f(\xi^*) = 0$ [4].

Based on the set of demonstrations, one can build an estimate $\hat{f}$ of f (we will further use the symbol $\hat{(\cdot)}$ to denote estimation values).

Definition 2 *$\hat{f}$ is a stable estimate of f in $\Re^d$ if 1) it has a single attractor ξ^*: $\hat{f}(\xi^*) = 0$ and 2) if any trajectory $\{\xi^t, \hat{\dot{\xi}}^t\}_{t=1}^T$, generated by $\hat{f}$ converges asymptotically to ξ^*:*

$$\lim_{t\to\infty} \xi^t = \xi^* \quad \forall \xi^t \in \Re^d \tag{2}$$

Non-linear dynamical systems are prone to instabilities. Ensuring that the estimate $\hat{f}$ results in *asymptotically stable trajectories*, i.e. trajectories that converge asymptotically to the attractor as per (2), is thus a key requirement for $\hat{f}$ to provide a useful control policy. A second key requirement is that the estimate follows closely the dynamics of the demonstrations, see Figure 1 for an example illustrating the above two desiderata. We evaluate the latter through a measure of the accuracy e with which $\hat{f}$ approximates the overall dynamics of the underlying model f. This can be quantified by measuring the discrepancy between the direction and amplitude of the estimated and observed velocity vectors for the T^n training points of each demonstration trajectory $\{\xi^{t,n}, \dot{\xi}^{t,n}\}_{n=1,t=1}^{N,T^n}$:

$$e(\dot{\xi}^n, \hat{\dot{\xi}}^n) = \left(\frac{1}{T^n} \sum_{t=1}^{T^n} r\left(\frac{1-\dot{\xi}^{t,n}\hat{\dot{\xi}}^{t,n}}{\|\dot{\xi}^{t,n}\|\|\hat{\dot{\xi}}^{t,n}\|+\epsilon}\right)^2 + q\frac{(\dot{\xi}^{t,n}-\hat{\dot{\xi}}^{t,n})^T(\dot{\xi}^{t,n}-\hat{\dot{\xi}}^{t,n})}{\|\dot{\xi}^{t,n}\|\|\hat{\dot{\xi}}^{t,n}\|+\epsilon} \right)^{\frac{1}{2}} \tag{3}$$

where r and q are positive scalars that weight the relative influence of each factor. ϵ is a very small positive scalar.

Note that the dynamics can be accurately estimated only in an area around the demonstration datapoints. Such an area is illustrated in green in Figure 1. Outside this region, lack of precise information regarding the original dynamics of the motion leads the system to follow a default dynamics that is asymptotically stable at the target.

A. Problem Formulation

To construct $\hat{f}$ from the set of demonstration trajectories $\{\xi^{t,n}, \dot{\xi}^{t,n}\}_{n=1,t=1}^{N,T^n}$, we follow a statistical approach and define $\hat{f}$ as a non-linear combination of a finite set of Gaussian functions.

Definition 3 *Consider a finite set of $k = 1..K$ Gaussians G^1 through G^K. Let μ^k and Σ^k be the mean and covariance matrix of Gaussian G^k. For a set of N demonstration trajectories $\{\xi^{t,n}, \dot{\xi}^{t,n}\}_{n=1,t=1}^{N,T^n}$, each point $[\xi^{t,n}, \dot{\xi}^{t,n}]$ is associated a probability $p(\xi^{t,n}, \dot{\xi}^{t,n})$:*

[4]Motions defined as per Eq. 1 should be used in conjunction with a low-level controller to compute the required force/torque for a robot to execute a motion trajectory.

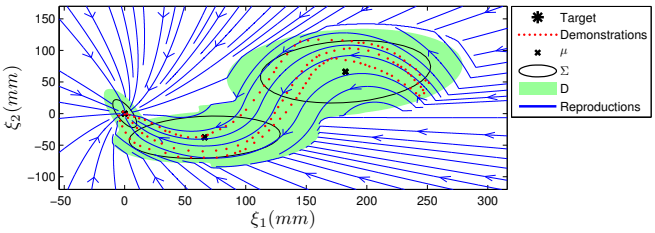

Fig. 1. Given a set of demonstrations, we build an estimate of the underlying dynamics of motion such that it is asymptotically stable at the target while following accurately a specific path with a particular dynamics. The green area highlight the region of the demonstrations, where the dynamics is known and followed accurately.

$$p(\xi^{t,n}, \dot{\xi}^{t,n}) = \frac{1}{K}\sum_{k=1}^{K} G^k(\xi^{t,n}, \dot{\xi}^{t,n}; \mu^k, \Sigma^k) \quad \begin{cases} \forall n \in 1..N \\ t \in 1..T^n \end{cases} \tag{4}$$

and

$$\mu^k = \begin{pmatrix} \mu^k_\xi \\ \mu^k_{\dot{\xi}} \end{pmatrix} \quad \& \quad \Sigma^k = \begin{pmatrix} \Sigma^k_\xi & \Sigma^k_{\xi\dot{\xi}} \\ \Sigma^k_{\dot{\xi}\xi} & \Sigma^k_{\dot{\xi}} \end{pmatrix} \tag{5}$$

The probability density function of the model $G^k(\xi^{t,n}, \dot{\xi}^{t,n}; \mu^k, \Sigma^k)$ is then given by:

$$G^k(\xi^{t,n}, \dot{\xi}^{t,n}; \mu^k, \Sigma^k) = \frac{1}{\sqrt{(2\pi)^d|\Sigma^k_\xi|}} e^{-\frac{1}{2}([\xi^{t,n},\dot{\xi}^{t,n}]-\mu^k)^T(\Sigma^k)^{-1}([\xi^{t,n},\dot{\xi}^{t,n}]-\mu^k)} \tag{6}$$

To generate a new trajectory from the Mixture of Gaussians, one can then sample from the probability distribution function given by Eq. 4, by computing the expectation on the posterior of the distribution, i.e. $\hat{\dot{\xi}} = \hat{f}(\xi) = E\{p(\dot{\xi}|\xi)\}$. This can be expressed as a weighted sum of *linear* dynamical systems given by:

$$\hat{\dot{\xi}} = \sum_{k=1}^{K} h^k(\xi)(A^k\xi + B^k) \tag{7}$$

where $A^k = \Sigma^k_{\dot{\xi}\xi}(\Sigma^k_\xi)^{-1}$, $B^k = \mu^k_{\dot{\xi}} - A^k\mu^k_\xi$, $h^k(\xi) = \frac{p(\xi;\mu^k_\xi,\Sigma^k_\xi)}{\sum_{k=1}^K p(\xi;\mu^k_\xi,\Sigma^k_\xi)}$, $h^k(\xi) > 0$, and $\sum_{k=1}^K h^k(\xi) = 1$.

The *non-linear* weighting terms h^k in Eq. (7) give a measure of the relative influence of each Gaussian locally. The projection through $A^k\xi + B^k$ is equivalent to performing a local linear fit onto data assigned to each Gaussian. Each weighting term h^k is inversely proportional to the variance of the data locally: the more variance (the less accurate the demonstrations), the less influence. Such a rewriting will prove useful to study the stability of the estimate as will be discussed in Section IV.

III. Building an Estimate of the Dynamics

Existing approaches to the statistical estimation of $\hat{f}$ use either Gaussian Process Regression (GPR) [10], Gaussian Mixture Regression (GMR) [4] or Locally Weighted Projection Regression (LWPR) [11]. GPR builds an accurate

estimate of non-linear functions, but the method is ill-suited for applications requiring fast computation because GPR's computational costs scale cubically with the number of training examples. LWPR in comparison offers a cost-efficient method for non-linear function approximation; see [12] for a comparison. LWPR describes the system through a finite combination of Gaussians. Parameters are estimated in one-shot learning through linear regression. LWPR, however, is very sensitive to the choice of parameters at initialization and relies on manual tuning to achieve high accuracy. Attempts at combining both approaches to achieve high-precision at bearable computational costs provide a promising route [13]. However, they remain computationally costly as the iteration steps increase linearly with the number of data points. Parametric techniques such as Gaussian Mixture Regression using Expectation Maximization learning (GMR-EM) provide an alternative method to modeling non-linear trajectories. GMR-EM usually requires much fewer parameters than the other mentioned methods, but is less accurate. Critical concerns with GMR-EM are that it requires a heuristic to find the optimal number of Gaussian kernels, and the final results are sensitive to initialization.

Irrespective of whether these methods are accurate, fast, etc., they cannot be used to estimate dynamical systems as per Eq. 1, mainly because they do not take into account the stability of the dynamical system they model[5]. They often result in unstable estimates of the function f. This is illustrated for GMR-EM and LWPR in Figure 2.

To address the instability issue, the Dynamic Motor Primitive (DMP) method [6] was proposed. DMP couples the estimate of $\hat{f}$ learned through Locally Weighted Regression (LWR)[6] with a stable *linear* dynamical system. Global stability is ensured by giving precedence to the linear dynamical system when approaching to the target, so that the dynamics are then dominated by the stable linear dynamical system. This method, however, has two drawbacks: First, modulation between the two dynamics (the dynamics inferred from the demonstrations and the linear DS) is done using a phase variable which decreases exponentially as time passes by. This phase variable makes the system time dependent and hence sensitive to temporal perturbations. For example, if a perturbation causes some delay in the execution time, this results in having a considerable error in the estimation (see Figure 2). Second, modeling multi-dimensional systems with DMP is done by learning one DS for each dimension separately, hence neglecting the combined effect of all the dimensions in the motion. As a result, a heuristic is required to synchronize the DS controlling for each dimension. The recent modifications on DMP [14] do not address these issues.

In this paper, we introduce an iterative method, Binary Merging (BM), for building a multi-dimensional Gaussian Mixture Model (GMM). The method minimizes the number of Gaussians required for achieving both asymptotic stability at the target and high accuracy in estimating the dynamics of motion (Figure 2). Next, we formalize the conditions for such a mixture to provide stable dynamics, as per Definition 2.

[5]GMR-EM and GPR optimize the likelihood that the complete model represents the data well. LWPR minimizes the mean-square error between the estimate and the data.

[6]LWR was used here because learning was performed on each dimension separately and thus did not require the local projection performed by LWPR.

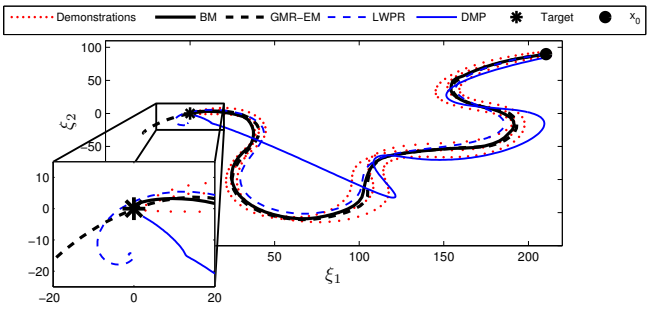

Fig. 2. An example of two-dimensional dynamics estimated on the basis of three training examples, using four different methods: BM, GMR-EM, LWPR, and DMP.

IV. Stability Analysis

We start by defining conditions for the estimate $\hat{f}$ to be *asymptotically stable* in $\Re^d$ and to accurately follow demonstrations within a region D of the state space (that covers entirely the part of the state space spanned by the demonstrations). We proceed by first determining a partition of D. We then define the effect of the Gaussian Mixture in each partition and determine conditions in each partition for ensuring stability in D. Then we extend these conditions to ensure global stability of the dynamics on $\Re^d$.

First, without loss of generality, we can assume that ξ^*, the unique stable attractor of f, is located at the origin, so that $f(\xi^*) = f(0) = 0$ and by extension $\hat{f}(\xi^*) = \hat{f}(0) = 0$. Let us then reverse the temporal ordering of datapoints in the demonstrations $\{\xi^{t,n}, \dot{\xi}^{t,n}\}_{n=1,t=1}^{N,T^n}$, such that $t = T^n$ refers to the onset of the motion and $t = 1$ to the end of the motion. The motion then evolves in time from $\xi^{T^n} \rightarrow \xi^t \rightarrow \xi^1$.

Let $D \subset \Re^d$ be the region where the demonstrated motion can be estimated accurately through $\hat{f}$:

Definition 4 *Consider a set of scalars $\delta^k \geq 0$, $k = 1..K$:*

$$\delta^k = \alpha \min_{\xi^{t,n} \in \Omega^k} (p(\xi^{t,n})) \tag{8}$$

where $p(\xi)$ is the probability of ξ estimated from Eq. 4 and $0 < \alpha \leq 1$ is a constant. Then the region D such that

$$D = \{ \xi \in \Re^d \ : \ p(\xi) \geq \delta^k \} \tag{9}$$

defines a partition of the space that comprises all training datapoints. $\tilde{D} = \Re^d \setminus D$ is the complement of D in $\Re^d$.

This definition of δ^k ensures that all training datapoints are included in D. As α decreases, the width of D increases; however, at the cost of depreciating the accuracy of the model over D.

To study the stability of $\hat{f}$, we partition D into K pairwise disjoint continuous subregions Ω^k via hyperplanes

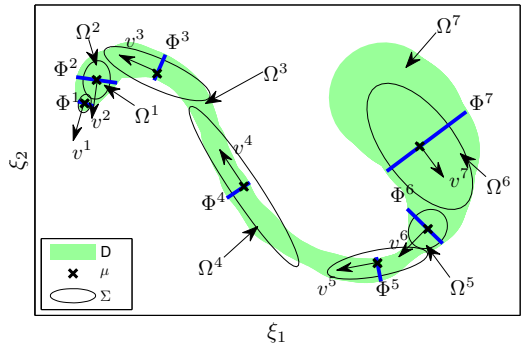

Fig. 3. Representation of subdomains $\Omega^k(\xi)$ for a sample 2-D model. Please see the text for further information.

Φ^k, $k = 1..K$ (see Appendix I). Each hyperplane Φ^k forms a boundary between the two adjacent subdomains Ω^k and Ω^{k+1} (see Figure 3). In each subdomain $\Omega^k \subset D$ we truncate the estimate given by Eq. 7 so that the dynamics are driven solely by the two dominant Gaussians G^k and G^{k+1}. Because $h^k(\xi)$ decays asymptotically as one moves away from the center of the associated Gaussian, the effect of truncating the influence of non-adjacent Gaussians is in practice negligible[7].

Since training is based on data covering only a subpart of the domain (the domain D), outside D the resulting estimate $\hat{f}$ of the dynamics may not have the same attractor landscape and basin of attraction as the original system, even if it approximates with high accuracy the original system locally[8]. In other words, outside D, the dynamics may be unstable and have local attractors that are different from the origin of the system. Thus, to ensure that $\hat{f}$ has a single attractor on $\Re^d$, outside D the value of $\hat{f}$ depends on the Gaussian G^1 given by Eq. 10-b. We will show later in this section how we can construct this Gaussian to ensure that ξ^* is the sole attractor of the system driven by Eq. 10-b.

$$\begin{cases} \textbf{(a-1):}\ \dot{\hat{\xi}} = \hat{f}(\xi) = h^{k+1}(\xi)(A^{k+1}\xi + B^{k+1}) + \\ \qquad +h^k(\xi)(A^k\xi + B^k) \quad \forall \xi \in \Omega^k\ \&\ k \in 1..K-1 \\ \textbf{(a-2):}\ \dot{\hat{\xi}} = \hat{f}(\xi) = A^K\xi + B^K \quad \forall \xi \in \Omega^K \\ \textbf{(b):}\quad \dot{\hat{\xi}} = \hat{f}(\xi) = A^1\xi \quad \forall \xi \in \tilde{D} \end{cases} \tag{10}$$

According to Eq. 10, the system follows the demonstrations accurately for all points $\xi \in D$ (i.e. Eq. 10-a). For points outside of D the system moves toward the target following solely the dynamics derived from the Gaussian G^1 (Eq. 10-b).

Theorem 1: Assume that the state trajectory evolves according to Eq. 10. Then the origin of Eq. 10 is asymptotically stable if the parameters of $\hat{f}$ (i.e. μ^k and Σ^k, $\forall k = 1..K$) are constructed such that

$$\begin{cases} \textbf{(a)}\ B^* = h^1(0)B^1 + h^2(0)B^2 = 0 \quad \xi = 0 \in \Omega^1 \\ \textbf{(b)}\ \begin{cases} (\xi - \mu_\xi^{k+1})^T(\Sigma_\xi^{k+1})^{-1}\dot{\hat{\xi}} > (\xi - \mu_\xi^k)^T(\Sigma_\xi^k)^{-1}\dot{\hat{\xi}} \\ \qquad \forall \xi \in \Omega^k\ \ \&\ \ \xi \neq 0\ \ \&\ \ \forall k \in 1..K-1 \\ (\xi - \mu_\xi^K)^T(\Sigma_\xi^K)^{-1}\dot{\hat{\xi}} < 0 \qquad \forall \xi \in \Omega^K \end{cases} \\ \textbf{(c)}\ (\upsilon^k)^T\dot{\hat{\xi}} > 0 \quad \forall \xi \in \Phi^k\ \ \&\ \ \xi \neq 0\ \ \&\ \ \forall k \in 1..K \\ \textbf{(d)}\ \text{starting from any point } \xi^0 \in D, \text{ the trajectory} \\ \quad \text{remains in D, i.e. } \{\xi^t\}_{t=0}^{\infty} \in D \\ \textbf{(e)}\ \Sigma^1_{\dot{\xi}\xi}(\Sigma^1_\xi)^{-1} \text{ is negative definite} \end{cases} \tag{11}$$

To elaborate more, condition **(a)** puts a constraint on Eq. 10 to force the origin to be an equilibrium point. Condition **(b)** defines criteria to ensure that starting from any point $\xi \in \Omega^k$, the energy of the system (i.e. Lyapunov Function) asymptotically decreases as motion evolves. Condition **(c)** ensures the transition of motions from one partition to another partition at the boundaries Φ^k (in this equation, υ^k correspond to the normal vector of the hyperplanes Φ^k). Condition **(d)** is necessary to avoid possible cyclic behavior in the system defined by Eq. 10. Putting together conditions **(a)-(d)**, the system becomes asymptotically stable at the origin for all trajectories inside D, i.e. Eq. 10-a. Condition **(e)** constructs the Gaussian G^1 such that all eigenvalues of A^1 become negative, and hence make $\hat{f}$ asymptotically stable at the origin for all points outside D, i.e. Eq. 10-b.

Proof: See Appendix II.

V. Learning Algorithm

Section IV provided us with conditions whereby the estimate, produced according to our state evolution paradigm (Eq. 10), is asymptotically stable at the origin in $\Re^d$. It remains now to determine a procedure by which we can construct a mixture of Gaussians to satisfy the conditions given in Eq. 11. Not only should the estimate be stable according to our earlier definition, but it should also provide an accurate estimate of the overall dynamics. Given a maximal accepted error e_{max}, estimates of the dynamics are accurate if

$$\bar{e} = \frac{1}{N}\sum_{n=1}^{N} e(\dot{\xi}^n, \dot{\hat{\xi}}^n) \leq e_{max} \tag{12}$$

where e is given by Eq. 3.

BM proceeds in two steps. First it initializes a model with the maximum possible number of Gaussians. Then it incrementally reduces the number of Gaussians to a minimum number (locally), which satisfies the stability criteria while keeping the error of the estimates below a certain level. Algorithm 1 shows the pseudocode of the BM procedure. Next, we briefly explain the main steps.

Initialization: First, demonstration trajectories are aligned using a *sample alignment method* [17]. These trajectories usually differ in length, as they may have been performed at

[7]This is especially true if the distance between the centers of the Gaussians is much larger than the variance of each Gaussian. Note that this formulation makes $\dot{\hat{\xi}}$ discontinuous at the boundaries between the subregions. This however does not affect the proof of Theorem 1, since the conditions Eq. 11 do not require continuity at the boundaries [15], [16].

[8]This is true irrespective of the technique (e.g. GMR, LWPR, GPR, etc) used for estimating $\hat{f}$.

different speeds. With sample alignment, demonstrations are aligned such that data points with the same time stamp have the most similarity based on a specified fitness function. The output of sample alignment is N demonstrations all of length T. Sample alignment differs from Dynamic Time Warping in that it does not distort the temporal transitions between datapoints in a demonstration. Figure 4-(a,b) illustrates the sample alignment procedure.

We use the time stamps that result from sample alignment to initialize the Gaussian mixture[9]. Let $\{\xi^{t,n}, \hat{\dot{\xi}}^{t,n}\}_{n=1,t=1}^{N,T}$ consists of the realigned set of demonstrated trajectories. We build $K = T$ Gaussians G^k with the time indices $t^k = k$ and $k = 1..K$. The mean and covariance of a Gaussian G^k is computed from a subset of the demonstrations $\{\xi^{t,n}, \dot{\xi}^{t,n}\}_{n=1,t=t^{k-1}+1}^{N,t^k}$ using:

$$\begin{cases} \mu^k = mean\left(\{\xi^{t,n}, \dot{\xi}^{t,n}\}_{n=1,t=t^{k-1}+1}^{N,t^k}\right) \\ \Sigma^k = cov\left(\{\xi^{t,n}, \dot{\xi}^{t,n}\}_{n=1,t=t^{k-1}+1}^{N,t^k}\right) \end{cases} \quad (13)$$

where $t^0 = 0 < t^1 < \cdots < t^K = T$. We will further use the format $G^k\left(\{\xi^t, \dot{\xi}^t\}_{t^{k-1}+1}^{t^k}\right)$ to represent such a construction. When $t^{k-1} + 1 = t^k$ we shorten the representation using $G^k\left(\{\xi^{t^k}, \dot{\xi^{t^k}}\}\right)$.

Iteration: Iteration proceeds as follows. A pair $\{G^k, G^{k+1}\}$ of adjacent Gaussians is picked at random and merged into a new Gaussian, by computing the new means and covariances on the union of data points associated to each of the two Gaussians using Eq. 13 (i.e. $G^k\left(\{\xi^t, \dot{\xi}^t\}_{t^{k-1}+1}^{t^{k+1}}\right)$). If the region defined by this pair satisfies the conditions in Eq. 11 and the error given by Eq. 12 does not increase, then the merged Gaussian replaces the two selected Gaussians. The new model is now composed of $K - 1$ Gaussians. The procedure terminates when it is no longer possible to merge any pair of Gaussians without decreasing the accuracy or becoming unstable. The model converges within a maximum $T(T-1)/2$ iterations[10]. Such a learning procedure results in a higher number of Gaussians along curvatures in the motion (e.g. observe that the straight parts in Figure 3 require fewer Gaussians than the highly curved parts). Note that BM does not ensure that the globally minimal number of Gaussians is obtained (the algorithm finds the local minimum), and several derivation of the model must be done to ensure that random sampling leads to a better solution than the initial one.

VI. Experimental Results

The algorithm was validated to control the point-to-point motions of two kinematically driven robots: a 6 degrees of freedom (DOF) Katana-T arm and the 4DOF right arm of the humanoid robot Hoap-3. We also further examine the method for 20 different dynamics through simulation data generated

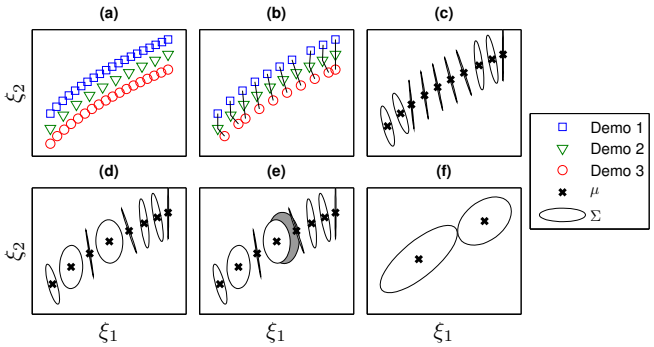

Fig. 4. Binary Merging (BM) learning algorithm. **(a)** Data points from three demonstrations. **(b)** Resultant trajectories after applying sample alignment. Points corresponding to the same time index are connected by a line. **(c)**Initialization step. **(d)** The GMM is updated by iteratively merging two pairs of adjacent Gaussians (2 with 3 and 5 with 6). **(e)** Because the new Gaussian resulting from the merging of 4 with 5 (shown in dark color) violates the stability criteria Eq. 11, the model remains unchanged. **(f)** Final model after termination.

using a Tablet PC. Training data was provided by a human expert 3-5 times for each example. In order to demonstrate the ability of the proposed algorithm to learn arbitrary dynamics, we chose tasks displaying characteristics typical of non-linear ODE dynamics with a single equilibrium point.

The first task, represented in Figure 5, consists of having the robot draw lines in a constrained 2D area. This task illustrates well the importance of having a stable controller that closely follows the learned dynamics. The second task requires the robot to move an object while avoiding an obstacle[11] (see Figure 6). The simulation results using a Tablet PC are also illustrated in Figure 7 (4 out of a total of 20 performed). In these tasks, inaccurate modeling may result in 1) a trajectory departed from the demonstrated one, 2) having a velocity profile different from the demonstrated one, or 3) the system converging to a spurious attractor.

We compared the performance of the proposed method (BM) against those of three alternative methods 1) LWPR, 2) GMR-EM, and 3) DMP. We tuned all initialization parameters of these methods so that they would, at least, follow the desired motion, while achieving the best possible performance in having the same velocity profile as the demonstrations. Quantitative performance comparisons are given in Table I. GMM with BM outperforms all other methods in that it is able to converge asymptotically to the target, while achieving comparable accuracy in following the dynamics, as shown in Table I. Moreover, it does so using a much smaller number of parameters than LWPR and DMP. Note that because GMR with EM requires a fixed set of Gaussians, we used the number of Gaussians found with BM at initialization. Similarly, the number of Gaussians used in DMP was initialized with that found with LWPR for the same task.

[9]The distortions resulting from aligning the trajectories are negligible if the sampling granularity is large.

[10]The maximum number of iteration happens when no merging is possible at the initialization. Thus the algorithm stops after examining the stability and accuracy criteria for all possible combinations of Gaussians.

[11]Currently BM does not explicitly consider the constraints of these examples in its learning procedure. However, BM is able to satisfy them by accurately following demonstrations in the domain D. However, if a trajectory starts outside D, it may violates the task's constraint(s).

TABLE I

PERFORMANCE COMPARISON OF THE MOTION GENERATION METHODS IN LEARNING 20 DIFFERENT POINT TO POINT MOTIONS

Method	GMR-BM	GMR-EM	LWPR	DMP
Average of $\bar{e}$	**0.1887**	**0.1538**	**0.2013**	1.6809
Range of $\bar{e}$	**0.1489 - 0.2537**	**0.1142 - 0.2179**	**0.1371 - 0.3796**	0.9893 - 4.8664
Average number of Gaussians	**8**	**8**	35	35
Range of number of Gaussians	**3 - 15**	**3 - 15**	8 - 62	8 - 62
The way the number of Gaussians is specified	**Automatic**	Manual	**Automatic**	Manual
Average number of parameters to encode the dynamics	**10*8 = 80**	11*8 = 88	21*35 = 735	2*(4+3*35)=218
Consider stability / Type of stability	**Yes / Global**	No / —	No / —	**Yes / Global**
Supporting multidimensional data	**Yes**	**Yes**	**Yes**	No
Adapt the model based on the dynamics complexity	**Yes**	**Yes**	No	No

Algorithm 1 Binary Merging (BM)

Input: $\{\xi^{t,n}, \dot{\xi}^{t,n}\}_{n=1,t=1}^{N,T^n}$, r, q, and e_{max}

Initialization
Transfer ξ^* to the origin.
Apply sample alignment to get N demonstrations, all of length T.
Reverse the order of the datapoints for all demonstrations.
Define $t^k = k, \forall k \in 1..T$
Create a GMM $\Gamma = \{G^1..G^K\}$ using $G^k\left(\{\xi^{t^k}, \dot{\xi^{t^k}}\}\right)$ as per Eq. 13.
Main Body
while $K > 1$ **&** further merging is possible **do**
  Backup the previous model $\tilde{\Gamma} \leftarrow \Gamma$
  Randomly select an index $k \in 1..K-1$
  Replace $G^k \rightarrow G^k\left(\{\xi^t, \dot{\xi}^t\}_{t^{k-1}+1}^{t^{k+1}}\right)$ by merging G^k and its adjacent G^{k+1}.
  Remove G^{k+1} and correct the numbering of Gaussians $G^{i-1} = G^i$ and time indices $t^{i-1} = t^i, \forall i \in k+1..K$
  $K \leftarrow K - 1$
  if conditions of Theorem I are violated **or if** Eq. 12 is no longer satisfied **then**
    Recover the previous model $\Gamma \leftarrow \tilde{\Gamma}$
    Set $K \leftarrow K + 1$
  end if
end while

Output: $\Gamma = \{G^1, ..., G^K\}$

VII. SUMMARY AND DISCUSSION

In this work, we presented the Binary Merging (BM) learning method for encoding point-to-point motions as a first order autonomous non-linear ODE. The underlying assumptions for BM are: 1) there is only one point with zero velocity in demonstrations, 2) all demonstrations are relevant, and 3) motions evolve according to Eq. 10. The main advantages of BM are: 1) The estimate of the underlying dynamical system is globally stable at the target. 2) It is robust to perturbations[12]. 3) The produced trajectories accurately follow the motion dynamics for all points within the region covered by demonstrations. 4) The number of required parameters are automatically set based on the motion complexity. 5) The number of tweaking parameters needed to initiate the learning procedure is small. 6) Fewer parameters are required compared with competitive methods. 7) The learning method is multi-dimensional and thus captures information about the correlation across state dimensions.

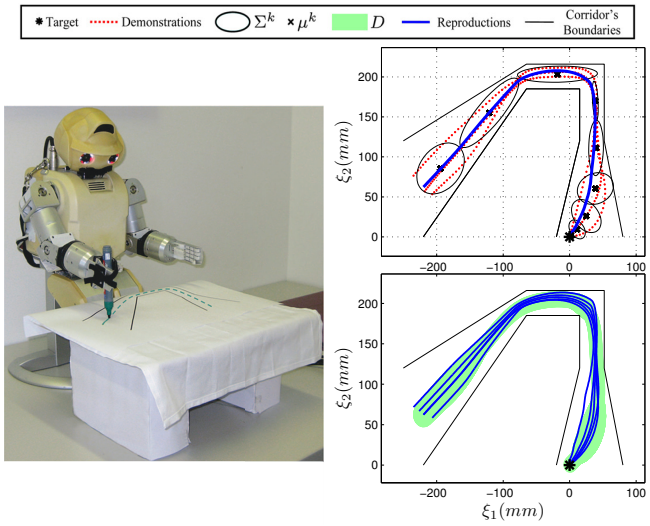

Fig. 5. The Hoap-3 robot performing the experiment of drawing lines in a constrained 2D area.

However, it is important to emphasize that while BM offers a relatively better performance than the other three methods, it does so at the cost of being much more computationally intensive than LWPR and DMP during training. While LWPR and DMP's training procedures increase with the dimension of the data and are of order $O(d)$ and $O(d^2)$ respectively [11], BM is of order $O(K * N * d)$. The complexity of BM is however small compared to GPR, but comparable to its novel extension [13] that increases linearly in the number of data points for both training and retrieval. Similarly to LWPR and DMP, and in contrast to GPR, BM's computational costs for the retrieval procedure are low and increase linearly with the number of parameters[13]. Since the Gaussian mixture trained with BM results in fewer parameters, the computation time is usually lower, which may be advantageous for real-time control.

[12]This is a property of all stable single attractor dynamical systems. When a perturbation disturbs the motion, the system automatically adapts itself to the new condition and converges to the target.

[13]GMR with Expectation-Maximization is unbounded. Each maximization step is however also of order $O(K * N * d)$.

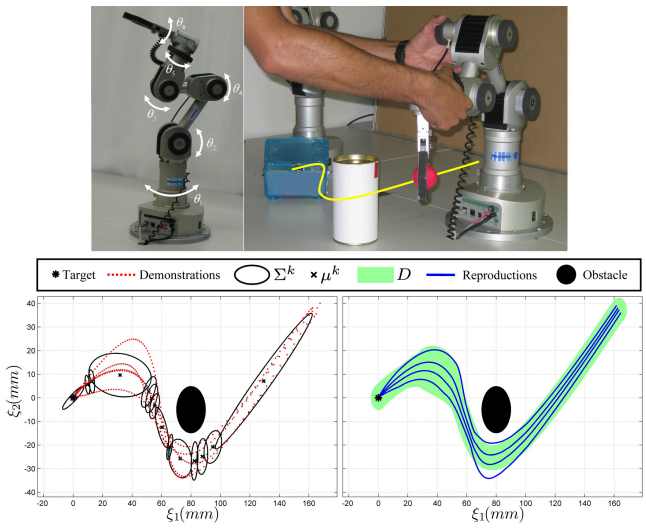

Fig. 6. The Katana-T arm performing the experiment of moving an object while avoiding an obstacle.

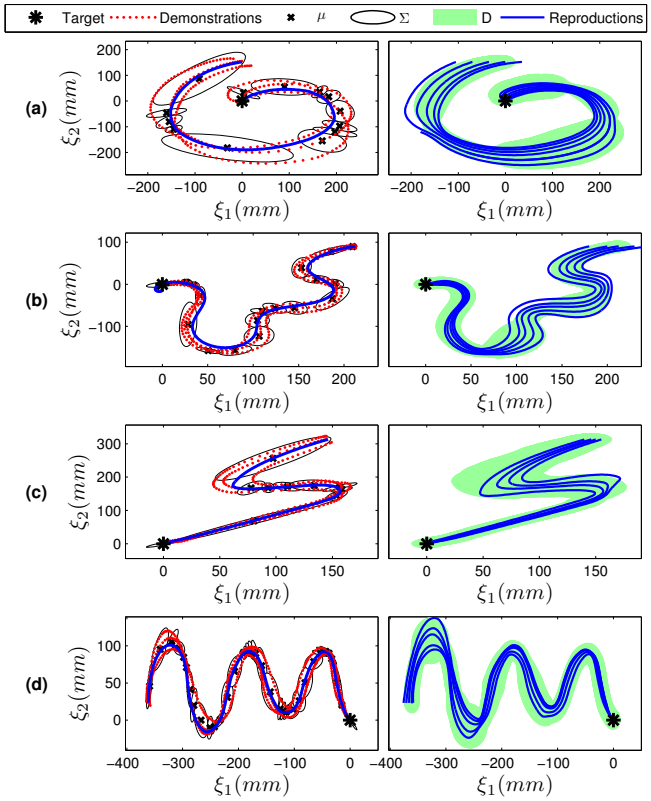

Fig. 7. Performance evaluation of learning different non-linear dynamics. Only 4 of 20 examples are shown here. Each row represents one motion.

An assumption made through this paper is that represented motions can be modeled with a first order time-invariant ODE. While the nonlinear function given by Eq. 10 is able to model a wide variety of motions, the method cannot be used for some special cases violating this assumption. Most of the time, this limitation can be tackled through a change of variable. For example a self-intersecting trajectory or a motion for which the starting and final points coincide with each other (e.g. a triangular motion) cannot be modeled through Eq. 10 if ξ codes solely for the end-effector cartesian position (i.e. $\xi = x \Rightarrow \dot{\xi} = \dot{x}$). But, if information about velocity is added (i.e. $\xi = [x; \dot{x}] \Rightarrow \dot{\xi} = [\dot{x}; \ddot{x}]$), the system can disambiguate the direction of motion at the intersection and hence successfully encode the dynamics of motion.

Furthermore it should be noted that while the stability condition given by Eq. 11-b should be verified for all points inside D, in practice due to non-linearity of $\hat{f}$ this verification can only be checked numerically on a mesh of datapoints defined over D [18]. The required granularity of the mesh depends on the level of complexity of a motion. Using a low granular mesh for highly non-linear function may result in error in evaluating the stability of $\hat{f}$. Thus it is necessary to tune the granularity of mesh such that it captures the non-linearity of the dynamics while keeping the computation time as small as possible.

As for future work, we are currently working extending BM to encode point-to-point motions for force/torque driven robots. This requires constructing the underlying dynamical system as a second order ODE. We will also focus on investigating a learning method that captures several different dynamical motions into one single model. This idea is inspired by human body motion: humans often follow different dynamics to execute a single task starting from different areas in the task space, depending on the whether he is considering his joint limits, the task constraints, or simply energy saving.

ACKNOWLEDGMENT

This work was supported by the European Commission through the EU Projects FEELIX-GROWING (FP6-IST-045169) and ROBOT@CWE (FP6-034002).

APPENDIX I
DEFINITION OF HYPERPLANES AND SUBREGIONS

Definition 5 *Consider a finite set of $k = 1..K$ Gaussians numbered G^1 through G^K. Let μ^k and Σ^k be the mean and covariance matrix of Gaussian G^k, as given by Eq. 5. Let the vector υ^k be the eigenvector of Σ^k_ξ forming the smallest angle with μ^k_ξ (i.e. υ^k is the eigenvector pointing towards the direction of motion). Then Φ^k is the hyperplane through μ^k_ξ and normal to υ^k:*

$$\Phi^k : \upsilon^k \cdot (\xi - \mu^k_\xi) = 0 \tag{14}$$

Definition 6 *The state-space domain D is partitioned into K pairwise disjoint continuous subregions Ω^k, $\bigcup_{k=1}^{K} \Omega^k = D$, $\Omega^k \cap \Omega^j = \emptyset$, $\forall k, j \in 1..K$ and $j \neq k$. Each subregion Ω^k is a part of D bounded by the hyperplanes Φ^k and Φ^{k+1}, s.t.*

$$\begin{cases} \hat{\Omega}^k = \{\, \xi \in D \;:\; v^{k+1} \cdot (\xi - \mu_\xi^{k+1}) > 0 \quad \& \\ \qquad\qquad v^k \cdot (\xi - \mu_\xi^k) \leq 0\} \quad \forall k \in 1..K-1 \\ \hat{\Omega}^K = \{\, \xi \in D \;:\; v^K \cdot (\xi - \mu_\xi^K) \leq 0\} \\ \Omega^k = \hat{\Omega}^k \bigcap D - \bigcup_{j=1, j\neq k}^{K} (\hat{\Omega}^k \cap \hat{\Omega}^j) \quad \forall k \in 1..K \end{cases} \tag{15}$$

APPENDIX II
PROOF OF THE THEOREM 1

Proof for Eq. 11-b: We start by recalling the following from [15], [16]. Suppose there exist a continuously differentiable non-linear system for which a piecewise Lyapunov function $V^k(\xi)$ is defined for each subdomain Ω^k. If $\forall \xi \in \Omega^k$ there exist positive constants α^k, β^k, and $s > 0$ such that:

$$\begin{cases} \textbf{(a):}\ \alpha^k \|\xi\|^s \leq V^k(\xi) \leq \beta^k \|\xi\|^s \quad \forall \xi \in \Omega^k \\ \textbf{(b):}\ \dot{V}^k(\xi) < 0 \quad \forall \xi \in \Omega^k \ \ \& \ \ \xi \neq 0 \\ \textbf{(c):}\ V^k(\xi) < V^{k+1}(\xi) \quad \forall \xi \in \Phi^k \end{cases} \tag{16}$$

then the origin is asymptotically stable in the sense of Lyapunov[14]. Consequently, given a system described by Eq. 10, and a positive scalar b^k, for every subregion Ω^k, we define a Lyapunov function $V^k(\xi)$ of the form:

$$\begin{cases} V^k(\xi) = \frac{E^{k+1}(\xi)}{E^k(\xi)} + b^k \quad \forall \xi \in \Omega^k \ \& \ k \in 1..K-1 \\ V^K(\xi) = \frac{1}{E^K(\xi)} + b^K \quad \forall \xi \in \Omega^K \end{cases} \tag{17}$$

where $E^k(\xi) = e^{-\frac{1}{2}(\xi - \mu_\xi^k)^T (\Sigma_\xi^k)^{-1} (\xi - \mu_\xi^k)}$, $\forall k \in 1..K$. Note that by construction $V^k(\xi)$ is positive, bounded, continuous and continuously differentiable in Ω^k, $\forall k = 1..K$. It can be easily shown that there always exist positive scalar α^k and β^k such that condition Eq. 16-a is satisfied. Similarly, one can find a set of positive scalar b^k to satisfy condition Eq. 16-c. Taking the derivative of V^k yields[15]:

$$\dot{V}^k(\xi, \dot{\xi}) = \frac{\dot{E}^{k+1}(\xi, \dot{\xi}) E^k(\xi) - E^{k+1}(\xi) \dot{E}^k(\xi, \dot{\xi})}{(E^k(\xi))^2} \tag{18}$$

$\dot{V}^k(\xi, \dot{\xi})$ (Eq. 18) is negative definite in each subregion Ω^k if and only if:

$$\begin{aligned} &\dot{V}^k(\xi, \dot{\xi}) < 0 \\ &\Leftrightarrow\ \dot{E}^{k+1}(\xi, \dot{\xi}) E^k(\xi) - E^{k+1}(\xi) \dot{E}^k(\xi, \dot{\xi}) < 0 \\ &\Leftrightarrow\ \frac{\dot{E}^{k+1}(\xi, \dot{\xi})}{E^{k+1}(\xi)} < \frac{\dot{E}^k(\xi, \dot{\xi})}{E^k(\xi)} \\ &\Leftrightarrow\ \frac{\frac{\partial E^{k+1}(\xi)}{\partial \xi} \dot{\xi}}{E^{k+1}(\xi)} < \frac{\frac{\partial E^k(\xi)}{\partial \xi} \dot{\xi}}{E^k(\xi)} \\ &\Leftrightarrow\ \frac{-(\xi - \mu_\xi^{k+1})^T (\Sigma_\xi^{k+1})^{-1} E^{k+1}(\xi) \dot{\xi}}{E^{k+1}(\xi)} < \frac{-(\xi - \mu_\xi^k)^T (\Sigma_\xi^k)^{-1} E^k(\xi) \dot{\xi}}{E^k(\xi)} \\ &\Leftrightarrow\ (\xi - \mu_\xi^{k+1})^T (\Sigma_\xi^{k+1})^{-1} \dot{\xi} > (\xi - \mu_\xi^k)^T (\Sigma_\xi^k)^{-1} \dot{\xi} \end{aligned}$$

Similarly, in Ω^K we have:

$$\dot{V}^K(\xi, \dot{\xi}) < 0 \quad \Leftrightarrow \quad 0 > (\xi - \mu_\xi^K)^T (\Sigma_\xi^K)^{-1} \dot{\xi}$$

which verifies conditions Eq. 11-b.

[14]Regarding the system described in Section IV, the only possible transitions are from subregions Ω^{k+1} to Ω^k via hyperplanes Φ^k (see Eq. 11-c), which results in having only one transition through each Φ^k.

[15]Note that both V and E are a function of ξ while their derivatives are a function of both ξ and $\dot{\xi}$.

Proof for Eq. 11-a: Let us assume that the origin (i.e. target) is the only point where $\dot{\hat{\xi}} = f(0) = 0$, and $0 \in \Omega^1$. Solving Eq. 10 for $\dot{\hat{\xi}} = 0$ and $\xi^* = 0 \in \Omega^1$ yields:

$$B^* = h^2(0) B^2 + h^1(0) B^1 = 0$$

Note that without having this condition, it is impossible to find appropriate α^k, β^k, and $s > 0$ to bound the Lyapunov function around the origin (recall $\xi^* = 0 \in \Omega^1$).

Proof for Eq. 11-e: Following from Eq. 7, observe that $A^1 = \Sigma_{\dot{\xi}\xi}^1 (\Sigma_\xi^1)^{-1}$. If A^1 is a negative definite matrix then all its eigenvalues are negative, and hence the dynamics driven by A^1 is asymptotically stable.

REFERENCES

[1] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233–242, 1999.

[2] D. Kulic, W. Takano, and Y. Nakamura, "Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden markov chains," *The International Journal of Robotics Research*, vol. 27(7), pp. 761–784, 2008.

[3] S. Calinon, F. Guenter, and A. Billard, "On learning, representing and generalizing a task in a humanoid robot," *IEEE transactions on systems, man and cybernetics*, vol. 37, no. 2, pp. 286–298, 2007.

[4] M. Hersch, F. Guenter, S. Calinon, and A. Billard, "Dynamical system modulation for robot learning via kinesthetic demonstrations," *IEEE Transactions on Robotics*, pp. 1463–1467, 2008.

[5] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation," *Philosophical Transactions: Biological Sciences (The Royal Society)*, no. 1431, pp. 537–547, 2003.

[6] J. A. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2002.

[7] D. Grimes, D. Rashid, and R. Rao, "Learning nonparametric models for probabilistic imitation," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, 2006.

[8] J.-H. Hwang, R. Arkin, and D.-S. Kwon, "Mobile robots at your fingertip: Bezier curve on-line trajectory generation for supervisory control," in *Proceedings of the IEEE/RSJ IROS*, 2003.

[9] R. Andersson, "Aggressive trajectory generator for a robot ping-pong player," *IEEE Control Systems Magazine*, vol. 9(2), pp. 15–21, 1989.

[10] C. Rasmussen and C. Williams, *Gaussian processes for machine learning*. Springer, 2006.

[11] S. Vijayakumar and S. Schaal, "Locally weighted projection regression: An o(n) algorithm for incremental real time learning in high dimensional space," in *Proceedings of the Seventeenth International Conference on Machine Learning (ICML)*, 2000.

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

[13] D. Nguyen-Tuong, M. Seeger, and J. Peters, "Local gaussian process regression for real time online model learning and control," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, 2008.

[14] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *International conference on robotics and automation (ICRA)*, 2009.

[15] S. Pettersson and B. Lennartson, "Exponential stability analysis of nonlinear systems using lmis," in *Proceedings of the 36th IEEE Conference on Decision and Control*, 1997.

[16] P. Borne and J.-Y. Dieulot, "Fuzzy systems and controllers: Lyapunov tools for a regionwise approach," *Nonlinear analysis*, pp. 653–665, 2005.

[17] C. Myers and L. Rabiner, "A comparative study of several dynamic time-warping algorithms for connected word recognition," *The Bell System Technical Journal*, vol. 60, no. 7, pp. 1389–1409, 1981.

[18] E. Gribovskaya, S. M. Khansari-Zadeh, and A. Billard, "Learning nonlinear multivariate dynamics of motion in robotic manipulators," *submitted*.