Learning the Nonlinear Multivariate Dynamics of Motion of Robotic Manipulators

E. Gribovskaya , S. M. Khansari Zadeh , Aude Billard

***Abstract*—Motion imitation requires reproduction of a dynamical signature of a movement, i.e. a robot should be able to encode and reproduce a particular path together with a specific velocity and/or an acceleration profile. Furthermore, a human provides only few demonstrations, that cannot cover all possible contexts in which the robot will need to reproduce the motion autonomously. Therefore, the encoding should be able to efficiently generalize knowledge by generating similar motions in unseen context.**

This work follows a recent trend in Programming by Demonstration in which the *dynamics* of the motion is learned. We present an algorithm to estimate multivariate robot motions through a Mixture of Gaussians.

The strengths of the proposed encoding are three-fold: i) it allows to generalize a motion to unseen context; ii) it provides fast on-line replanning of the motion in the face of spatio-temporal perturbations; iii) it may embed different types of dynamics, governed by different attractors.

The generality of the method to estimate arbitrary non-linear motion dynamics is demonstrated by accurately estimating a set of known non-linear dynamical systems. The platform-independency and real-time performance of the method are further validated to learn the non-linear dynamics of motion in an industrial six degree of freedom robotic arm and in a four degree of freedom humanoid arm.

***Index Terms*—Non-Linear Autonomous Dynamical Systems Robot Programming by Demonstration Learning by Imitation Gaussian Mixture Model and Regression**

I. Introduction

The versatility of tasks that modern robots should accomplish has forced researchers to consider alternative methods for control. Designing task- and robot-specific controllers seems nowadays a time-consuming and ineffective solution, and preference gradually changes in favor of flexible and generic control methods that can adapt to various tasks and robots' geometries. If, in addition, the robot is expected to operate in the vicinity of or in collaboration with unskilled human users, control must be both intuitive and flexible to ensure safe and easy operability by the human.

Programming by Demonstration (PbD) has appeared as one way to respond to this growing need for intuitive control methods [Billard et al., 2008]. PbD designs user-friendly methods by which a human teaches a robot how to accomplish a given task, simply by demonstrating this task. One of the requirements for such a teaching method to be effective is that the number of training examples should remain small (one considers between five and ten examples to be a bearable number for the trainer). Consequently, PbD either relies on prior knowledge to speed up learning, or results in a partial representation of the task which can be refined later.

PbD operates at different levels of the task representation: from copying low-level features of the motion [Sternad and Schaal, 1999, Ude et al., 2004, Calinon and Billard, 2008, Nguyen-Tuong et al., 2008, Schaal et al., 2003] to inferring the user's intention using a symbolic representation [Demiris and B.Khadhouri, 2006, Zollner et al., 2004]. In this paper, we focus on a low-level representation of motions, therefore we further review work related to this direction of PbD. Low-level representations should determine the encoding of the demonstrated trajectories of motion so that they can be easily modulated to enable re-use of the skill in novel contexts. An overview of requirements for effective movement encoding has been summarized in [Ijspeert et al., 2001].

Most relevant to the present paper are the notions of *compactness* and *reusability* of the representation, i.e. the encoding should be easily transferrable to related tasks, and the notion of *robustness* to perturbations, i.e. an ability of an encoding to ensure that a motion may be quickly adapted to perturbation and changes in a dynamic environment.

Dynamical Systems (DS) provide an effective and elegant means of encoding motions, that fulfills the above three criteria. DS encode trajectories through a time-independent function that defines the temporal evolution of the motion. Generalization of the motion to an unobserved part of the space results immediately from the application of the function to the new set of input variables.

In this paper, we consider the problem of estimating a *time-independent* model of motion through a set of first order non-linear multivariate dynamical systems. We exploit the strength of parametric statistical techniques to learn correlations across the variables of the system and show that this technique allows the determination of a coarse representation of the dynamics. We demonstrate advantages of such an approach as an alternative to the *time-dependent* methods, by ensuring robustness to external spatio-temporal perturbations through on-line adaptation of the motion.

This paper is divided as follows. Section II reviews related work on motion learning and estimation of dynamical systems. Section III-A starts with a formalization of the problem at hand and the particular approach of this work. This is followed by a technical description of the modeling approach: Section III-B introduces the learning approach to estimate the dynamics, while Section III-C presents an iterative algorithm to improve stability of the learned dynamics. Finally, in Section IV, we validate the method by estimating the motion dynamics from trajectories generated with given dynamical laws; in this way we may systematically verify approximation qualities of the

method. We, further, show how the same framework can be used to learn the dynamics of motion of a 4 degree of freedom humanoid robot arm and a 6 degree of freedom industrial arm. The legend used in graphs throughout the paper is summarized in Figure 1. The glossary is in Table I.

II. Related Work

To better delineate this paper's particular contribution to both machine learning and robotics, we focus our review on two major themes. First, to situate the dynamical systems approach taken in our work, we make a brief historical tour of the large volume of literature on modeling robot motion, contrasting time-dependent and time-independent representations. We then turn to the problem of estimating arbitrary dynamical systems and introduce the particular statistical technique used here. We briefly summarize the broad division across parametric and non-parametric statistical methods, and situate our choice of parametric method in this context.

A. Motion Learning

A core issue within robot control is ensuring that, if perturbed, the robot's motion can be rapidly and on-the-fly recomputed to ensure that the robot ultimately accomplishes the task at hand. Perturbations may lead the robot to either depart from its original trajectory (e.g. when slipping or hitting an object) or be delayed (e.g. when slowed down because of friction in the gears). In the rest of this paper, we will refer to the former type of perturbations as *spatial* perturbations and to the latter as *temporal* perturbations.

The vast majority of work on motion learning has addressed essentially the problem of being robust to spatial perturbation. Very little work has been yet done on handling temporal perturbations, which is core to the model we develop here. Next, we review these different approaches.

B. Time-dependent Modeling Approaches

Traditional means of encoding trajectories are based on spline decomposition after averaging across training trajectories [Hwang et al., 2003, Andersson, 1989, Yamane et al., 2004, Aleotti et al., 2005]. Spline decomposition remains a powerful tool for quick trajectory formation. It is, however, heavily dependent on a heuristic for segmenting and aligning the trajectories. Furthermore, spline representation, not being statistically-based, may have difficulties in coping with noise in data that is inherent in the robotic application.

Non-linear regression techniques were proposed as a statistical alternative to spline-based representation [Calinon et al., 2007, Schaal and Atkeson, 1998, 1994, D. et al., 2008]. These methods allow the systematical treatment of uncertainty by assuming the noise in data and, therefore, by estimating actual trajectories as a set of random variables with learned parameters.

However, similarly to spline-based approaches, regression techniques depend on an explicit time-indexing and virtually operate in "open-loop". The lack of any kind of feedback makes regressions sensitive to both temporal and spatial perturbations. To compensate for this, one needs to introduce an external mechanism to track potential deviations from the desired trajectory during reproduction. Adaptation to deviations then relies on a heuristic to re-index the new trajectory in time or extrapolate in space. Such re-indexing or extrapolation often comes at the cost of deviating importantly from the desired velocity and acceleration profile, making the motion look "unnatural". Furthermore, finding a good heuristic is highly task-dependent and becomes particularly not-intuitive in multidimensional spaces [Schaal et al., 2003].

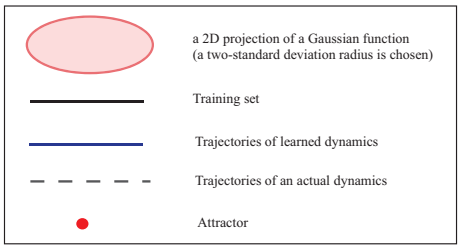

Fig. 1. Legend for the Figures in the paper.

Time-independent models, such as autonomous dynamical systems (to which we will further refer to as DS), were recently advocated as an alternative to the above approaches[1]. Models based on DS are advantageous in that they do not depend on an explicit time-indexing and thus provide a closed-loop controller, while being able to model arbitrary non-linear dynamics. Removing the explicit time-dependency comes at a cost, as it re-introduced an old problem, namely the need to consider stability of the control policy.

Next, we review current approaches to DS modeling of robot motion and point out the limitations of these methods. For a detailed discussion on advantages and disadvantages of dynamical systems encoding of motion, see also [Ijspeert et al., 2001, Schaal et al., 2003, 2001, Schoner and Santos, 2001].

C. Dynamical Systems Modeling of Motion

A number of recent approaches in PbD, including our prior work, investigate the use of dynamical systems for modeling robot motions [Ijspeert et al., 2001, Righetti et al., 2006, Dixon and Khosla, 2004, Ijspeert and Crespi, 2007, Hersch et al., 2008]. While [Dixon and Khosla, 2004] focuses on fitting the parameters of a first-order linear dynamical system into training data, the other above works tackle a problem of modulating a predefined linear dynamics with a non-linear estimate of a trajectory [Hersch et al., 2008] or a velocity profile [Ijspeert et al., 2001, Righetti et al., 2006]. The authors choose an uni-variate spring and damper system as an underlying linear dynamics. In such a way, they avoid an issue of stability of approximation that may occur if one learns an actual dynamics from data. However, this solution comes with its drawbacks: (1) uni-variate encoding discards information about correlation between degrees of freedom, that may be crucial for faithful reproduction (see Figure 23 for

[1]DS formulation embeds the time-dependency of a system in the mathematical formulation of the problem by using time derivatives of the state variables.

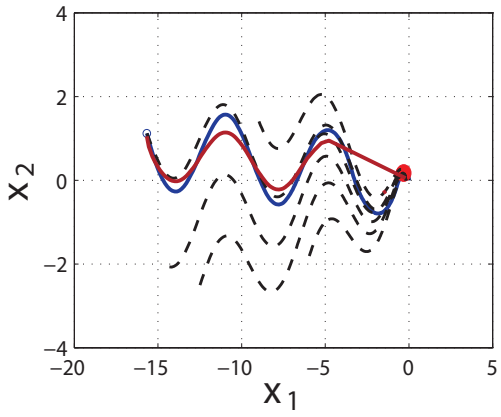

Fig. 2. A two-dimensional theoretical dynamical system is estimated from five training samples (dotted lines). using Dynamical Motion Primitives [Ijspeert et al., 2002](red solid line) and the method proposed in this paper (blue solid line). As approximation with Dynamical Motion Primitives requires combining statistical results with a predefined dynamics, it may deform an actual path to follow, as it can be seen on the graph.

illustration of the uni-variate encoding problem). (2) Coupling of the output of a predefined linear DS with a regression estimate makes the overall system dependent on the temporal synchronization between the two signals and thus in effect time-dependent (see Table VI for a formal comparison between the proposed approach and the work [Ijspeert et al., 2001]). To handle temporal perturbations, one would need a heuristic to maintain the synchronization. This would, however, no longer guarantee that the overall system is globally asymptotically stable. (3) By ensuring that the stable DS takes precedence over the estimate when coming close to the attractor or after a given time period, one can show global stability of the complete estimate [Ijspeert et al., 2002]. In effect, the global dynamics of motion is increasingly dominated by the stable linear dynamical system, hence leading the motion to progressively depart from the learned dynamics. This effect is illustrated in Figure 2, where we see that the trajectory is distorted as the system approaches the target. To ensure that the modulation still influences the dynamics of the motion when approaching the target, the method relies on using a large number of Gaussians spread across the data points.

In this paper, we develop an iterative procedure to learn a statistical estimate of an arbitrary multivariate autonomous dynamical system. We discuss the problem of stability of a learned estimate and propose an empirical procedure to verify stability and the region of applicability of the estimate. This relieves us from the need of using another a priori stable dynamical system and ensures robustness against spatial and temporal perturbations.

D. Estimating a Dynamical System

Data-driven methods for estimating dynamical systems consider multivariate input-output data as instances of a dynamical system and seek an estimate of the model that relates best these pairs of datapoints. Building a local approximation of the dynamics has been first reviewed within the time series analysis [Priestley, 1980, Chamroukhi et al., 2009, Ljung, 2004]. These works consider solely uni-dimensional data with a major motivation of predicting time series.

Analysis of dynamics has gradually shifted to state-space representation as it allows a representation of more sophisticated phenomena [Aoki, 1990, Crutchfield J.P., 1987]. The vast majority of these works focus on estimating *linear* dynamics [Dixon and Khosla, 2004, Ryoung K. Lim, 1998], a restrictive assumption for robotic applications. Recently, with the growing interest in chaos theory, more developed approaches have been proposed that allow approximation of complex dynamics [Crutchfield J.P., 1987, Wang et al., 2008]. While, several optimistic results in simulations have been presented [Carroll, 2007, Xie and Leung, 2005], their applicability to practical tasks with a small number of observed data containing noise remains to be verified.

The major body of numerical approaches of non-linear dynamical systems perform function approximation using different orthogonal polynomials (Chebyshev polynomials, B-splines [Lee, 1986], Radial Basis Functions [Buhmann, 2003] (RBFs)). Recently, many works have addressed the approximating properties of RBFs [Tomohisa et al., 2008, Travis et al., 2009, Wei and Amari, 2008]. RBFs have been proved to form universal approximators of any function on a compact set [Park and Sandberg, 1991]: any level of precision of the approximation may be achieved by considering an exhaustive number of basis functions; however, the quality of the approximation heavily depends on tuning a considerable amount of parameters. Thus, the problem of determining a tuning procedures optimum according to different criteria is a recurrent subject in the domain [Buhmann, 2003]. Furthermore, as the approximation with RBFs falls naturally into the category of non-parametrical methods discussed next, they suffer from the same types of limitations: RBFs better suits for approximation of uni-variate signals and quality of approximation rapidly deteriorates with an increase in the number of dimensions.

E. Statistical Encoding

Classically, the whole body of statical methods can be broadly divided into *parametric* and *non-parametric* approaches.

Non-parametric methods used in robot motion estimation include k-nearest neighbors [Moore, 1990], Gaussian Processes [Deisenroth et al., 2009, Nguyen-Tuong et al., 2008], Locally Weighted Regression, [Hardle, 1991, Muller, 1988, Schaal and Atkeson, 1998, 1994] and a combination of these [Nguyen-Tuong et al., 2008]. Non-parametric methods are advantageous over parametric methods as they make little assumptions about the form of the underlying distribution function to estimate. Moreover, due to the local nature of their estimate, non-parametric methods are well suited for accurate data fitting in low-dimensional spaces [Schaal and Atkeson, 1998, 1994]. Initially proposed for uni-dimensional problems, the above non-parametrical methods suffer from the curse of dimensionality [Bellman, 1957]: sparsity of training data in high-dimensional spaces makes accurate estimation of parameters almost impossible. Parametric methods, in contrast, are

better suited to model a multivariate dataset. They, however, rely on heuristics to choose the underlying parameters efficiently.

The Gaussian Mixture Models (GMMs) and based on them Gaussian Mixture Regression (GMR) are parametric methods. They are thus better suited for regression on multi-dimensional data [Sung, 2004]. Learning with GMM is classically done using Expectation-Maximization (EM), the iterative algorithm that optimizes the likelihood of the mixture of Gaussians over the data. Optimal performance relies, however, on choosing the number of Gaussians and on the stopping criterion of EM (see [McLahlan and Peel, 2000] for a review). While several methods have been proposed to automatically estimate these two parameters, with the Bayesian Information Criterion (BIC[2]) being the most generic, GMM estimation using EM may lead to suboptimal results and remain very sensitive to the initialization conditions. Here, we show that, for both our problem at hand and in practice, these known limitations are not an impediment and that an iterative method for choosing the number of Gaussians leads to good performance. Most importantly, we show that the method converges quickly and relies on very few parameters in comparison to parametric methods.

III. Method

A. Problem Statement

Consider that the state[3] of our robotic system can be unambiguously described by a variable ξ and that the workspace of the robot forms a sub-space X in $\mathbb{R}^N$.

Consider further that the state of our robotic system is governed by an *Autonomous Dynamical System* $\langle \mathcal{X}, f, \mathcal{T} \rangle$ (as per Definition 1-2, Table I). Then, for all starting locations $\xi_0 \in X$, the temporal evolution of our robotic system is *uniquely determined* by the *state transition map* (Definition 2, Table I) $f(t, t_0, \xi_0) = \xi(t)$, $\forall \xi_0, \xi \in X$.

Let us further assume that the state transition map f is a non-linear continuous and continuously differentiable function and that the system is driven by a first order differential equation[4] with a single equilibrium point $\bar{\xi}$, such that:

$$\forall t \in T = [t_0; \infty]; [\xi; \dot{\xi}] \in X \subset \mathbb{R}^N \tag{1}$$

$$\dot{\xi}(t) = f(\xi(t)) \tag{2}$$

$$\dot{\bar{\xi}} = f(\bar{\xi}) = 0. \tag{3}$$

Let the set of M N-dimensional demonstrated datapoints $\{\xi^i, \dot{\xi}^i\}_{i=1}^M$ be instances of the above motion model. The problem consists then of building an estimate $\hat{f}$ of f based on the set of demonstrations. To this end, we will approximate the function in a subregion[5] $C \subset X$, so that:

$$\hat{f} : C \to C \tag{4}$$
$$\hat{f}(\xi(t)) \cong f(\xi(t)), \forall \xi \in C.$$

C is further referred to as the *region of applicability* of a learned dynamics.

Without loss of generality, we can transfer the attractor to the origin[6], so that $\bar{\xi} = 0 \in C \subset X$ is now the equilibrium point of f and by extension of its estimate $\hat{f}$, i.e. $\hat{f}(0) = f(0) = 0$. If C is contained within the *region of attraction* Δ of $\bar{\xi}$ (see Definition 7, Table I), then the estimate $\hat{f}$ is asymptotically stable at $\bar{\xi}$ in C and any motion initiated from $\xi(t_0) \in C$ will asymptotically converge to the target $\bar{\xi}$.

TABLE I
Glossary of definitions

Definition 1: The *state-space* $X \subset \mathbb{R}^N$ includes all possible instantiations of ξ, such that $\xi(t) \in X$ at each time step $t \in T = \mathbb{R}^+ = [0; \infty]$.

Definition 2: A *dynamical system* is the tuple $\langle \mathcal{X}, f, \mathcal{T} \rangle$, with $f : t \to f^t$ a continuous map of X onto itself.

Definition 3: A dynamical system is *differentiable* if $\exists f : T \times X \to X$ such that for all $t_0 \in T, \xi_0 \in X$ the problem:

$$\dot{\xi} = f(t, \xi(t)), \;\; t \geq t_0, t \in T$$

$$\xi(t_0) = \xi_0$$

has a unique solution.

A dynamical system governed by a time-independent transition map with $f(t, \xi(t)) \triangleq f(\xi(t))$ is an *Autonomous Dynamical System*.

Definition 4. An *equilibrium state* $\bar{\xi} \in X$ of a dynamical system is such that

$$f(t, t_0, \bar{\xi}) = \bar{\xi}.$$

Definition 5. An equilibrium state $\bar{\xi} \in X$ is *stable* if $\exists \epsilon > 0$ and $\delta = \delta(\epsilon)$ such that

$$\forall \xi_0 \in B(\bar{\xi}, \delta) \Rightarrow f(\xi_0) \in B(\bar{\xi}, \epsilon),$$

$B(\bar{\xi}, \delta) \subset X$ is a hypersphere centered at $\bar{\xi}$ with radius δ. $\bar{\xi}$ is an *attractor* of f.

Definition 5. An *attractive state* is an equilibrium state $\bar{\xi}$ of a local flow, if there exists $\rho > 0$ such that:

$$\forall \xi_0 \in B(\bar{\xi}, \rho) \Rightarrow \lim_{t \to \infty} f(\xi_0) = \bar{\xi}.$$

$B(\bar{\xi}, \delta) \subset X$ is a hypersphere centered at $\bar{\xi}$ with radius δ. $\bar{\xi}$ is an *attractor* of f.

Definition 6. An equilibrium point $\bar{\xi}$ is *asymptotically stable* if it is both stable and attractive.

Definition 7. A set $\Delta \subset X$ is a *Region of Attraction (or Basin of Attraction)* of an equilibrium $\bar{\xi}$ if:

$$\Delta(\bar{\xi}) = \{\xi_0 \in X; \lim_{t \to \infty} f(\xi_0) = \bar{\xi}\}$$

See Figure 23-II for illustration.

Definition 8. A dynamical system is *globally asymptotically stable* at the equilibrium $\bar{\xi}$ if $\bar{\xi}$ is an asymptotically stable attractor and $\Delta \equiv X$.

[2]BIC introduces a penalty term for increasing the number of parameters in the model over the resulting improvement in the modeling performance.

[3]The state of a dynamical system represents the *minimum amount of information* required to describe the effect of past history on the future development of this system [Hinrichsen D., 2000].

[4]Considering solely *first order* dynamical systems is not restrictive to learning only first order relationships between trajectory and velocity, as one can always convert dynamics of an arbitrary order into a canonical system of first order ODEs.

[5]Estimating the dynamics in the whole state-space X would be practically infeasible due to the excessive number of demonstrations that this would require.

[6]To simplify the notation, we keep the same notation for the domains C and X after translation at the origin.

B. Approximating the Dynamics with Gaussian Mixture Regression

To construct $\hat{f}$ from the set of demonstrated trajectories, we follow a statistical approach and define $\hat{f}$ as a non-linear combination of a finite set of Gaussian kernels, using *Gaussian Mixture Models* (GMM).

GMMs define a joint probability distribution function $\mathcal{P}(\xi^i, \dot{\xi}^i)$ over training set of demonstrated trajectories $\{\xi^i, \dot{\xi}^i\}$, $i = 1..M$, M is the number of demonstrations, as a mixture of a finite set of K Gaussians $G^1..G^K$ (with μ^k and Σ^k being the mean value and covariance matrix of a Gaussian G^k):

$$\mathcal{P}(\xi^i, \dot{\xi}^i) = \frac{1}{K} \sum_{k=1}^{K} G^k(\xi^i, \dot{\xi}^i; \mu^k, \Sigma^k) \tag{5}$$

and

$$\mu^k = [\mu^k_\xi \; ; \; \mu^k_{\dot{\xi}}] \text{ and } \Sigma^k = \begin{pmatrix} \Sigma^k_\xi & \Sigma^k_{\xi\dot{\xi}} \\ \Sigma^k_{\dot{\xi}\xi} & \Sigma^k_{\dot{\xi}} \end{pmatrix} \tag{6}$$

Where each Gaussian probability distribution G^k is given by:

$$G^k(\xi^i_t, \dot{\xi}^i_t; \mu^k, \Sigma^k) = \frac{1}{\sqrt{(2\pi)^{2d}|\Sigma^k|}} e^{-\frac{1}{2}(([\xi^i_t, \dot{\xi}^i_t] - \mu^k)^T (\Sigma^k)^{-1} ([\xi^i_t, \dot{\xi}^i_t] - \mu^k))}. \tag{7}$$

The model is initialized using the *k-means* clustering algorithm starting from a uniform mesh and refined iteratively through Expectation-Maximization (EM) [Dempster et al., 1977].

To generate a new trajectory from learned GMMs, one can then sample from the probability distribution function given by Eq.5. This process is called Gaussian Mixture Regression (GMR).

Taking the posterior mean estimate of $\mathcal{P}(\dot{\xi}|\xi)$, the estimate of our function $\hat{\dot{\xi}} = \hat{f}(\xi)$ can then be expressed as a non-linear sum of linear dynamical systems, given by:

$$\hat{\dot{\xi}} = \sum_{k=1}^{K} h_k(\xi)(A_k \xi + B_k), \tag{9}$$

where $A_k = \Sigma^k_{\dot{\xi}\xi}(\Sigma^k_\xi)^{-1}$, $B_k = \mu^k_{\dot{\xi}} - A_k \mu^k_\xi$, $h_k(\xi) = \frac{\mathcal{P}(\xi; \mu^k_\xi, \Sigma^k_\xi)}{\sum_{k=1}^{K} \mathcal{P}(\xi; \mu^k_\xi, \Sigma^k_\xi)}$, $h_k(\xi) > 0$, and $\sum_{k=1}^{K} h_k(\xi) = 1$.

Such a rewriting will prove useful when studying the stability of the estimate, as will be discussed in Section III-C.

A geometric illustration of the GMR inference in the case of single Gaussian is presented in Figure 3 and the GMR procedure is summarized in Table II. Figure 4 further illustrates the encoding process from GMM to GMR for a non-linear dynamical system with a single attractor.

TABLE II
GAUSSIAN MIXTURE REGRESSION

Let us assume that we can for each input datapoint $\xi^{\mathcal{I}}$ match an output datapoint $\xi^{\mathcal{O}}$, the joint probability of input and output data is then modeled using Gaussian Mixtures. The probability that a datapoint $\eta = [\xi^{\mathcal{O}}; \xi^{\mathcal{I}}]$ belongs to the GMM is defined by

$$\mathcal{P}(\eta) = \sum_{k=1}^{K} \pi_k \, \mathcal{N}(\eta; \mu_k, \Sigma_k) =$$

$$= \sum_{k=1}^{K} \pi_k \, \frac{1}{\sqrt{(2\pi)^D |\Sigma_k|}} \, e^{-\frac{1}{2}\left((\eta - \mu_k)^\top \Sigma_k^{-1} (\eta - \mu_k)\right)}$$

where π_k are prior probabilities and $\mathcal{N}(\mu_k, \Sigma_k)$ are Gaussian distributions defined by centers μ_k and covariance matrices Σ_k, where input and outputs components are represented separately as

$$\mu_k = \begin{bmatrix} \mu_k^{\mathcal{I}} \\ \mu_k^{\mathcal{O}} \end{bmatrix}, \quad \Sigma_k = \begin{bmatrix} \Sigma_k^{\mathcal{I}} & \Sigma_k^{\mathcal{I}\mathcal{O}} \\ \Sigma_k^{\mathcal{O}\mathcal{I}} & \Sigma_k^{\mathcal{O}} \end{bmatrix}.$$

Gaussian Mixture Regression allows to compute for a given input variable $\xi^{\mathcal{I}}$ and a given component k, the expected distribution of $\xi^{\mathcal{O}}$ as:

$\mathcal{P}(\xi^{\mathcal{O}}|\xi^{\mathcal{I}}, k) \sim \mathcal{N}(\hat{\eta}_k, \hat{\Sigma}_k)$, where

$$\hat{\eta}_k = \mu_k^{\mathcal{O}} + \Sigma_k^{\mathcal{O}\mathcal{I}} (\Sigma_k^{\mathcal{I}})^{-1} (\xi^{\mathcal{I}} - \mu_k^{\mathcal{I}}),$$
$$\hat{\Sigma}_k = \Sigma_k^{\mathcal{O}} - \Sigma_k^{\mathcal{O}\mathcal{I}} (\Sigma_k^{\mathcal{I}})^{-1} \Sigma_k^{\mathcal{I}\mathcal{O}}.$$

where $h_k = \mathcal{P}(k|\xi^{\mathcal{I}})$ is the probability of the component k to be responsible for $\xi^{\mathcal{I}}$

$$h_k = \frac{\mathcal{P}(k)\mathcal{P}(\xi^{\mathcal{I}}|k)}{\sum_{i=1}^{K} \mathcal{P}(i)\mathcal{P}(\xi^{\mathcal{I}}|i)} = \frac{\pi_k \, \mathcal{N}(\xi^{\mathcal{I}}; \mu_k^{\mathcal{I}}, \Sigma_k^{\mathcal{I}})}{\sum_{i=1}^{K} \pi_i \, \mathcal{N}(\xi^{\mathcal{I}}; \mu_i^{\mathcal{I}}, \Sigma_i^{\mathcal{I}})}.$$

Alternatively, by using the linear transformation property of Gaussian distributions, the conditional expectation of $\xi^{\mathcal{O}}$ given $\xi^{\mathcal{I}}$ can be defined approximately defined by a single normal distribution with the parameters:

$$\hat{\mu} = \sum_{k=1}^{K} h_k \, \hat{\mu}_k \, , \quad \hat{\Sigma} = \sum_{k=1}^{K} h_k^2 \, \hat{\Sigma}_k. \tag{8}$$

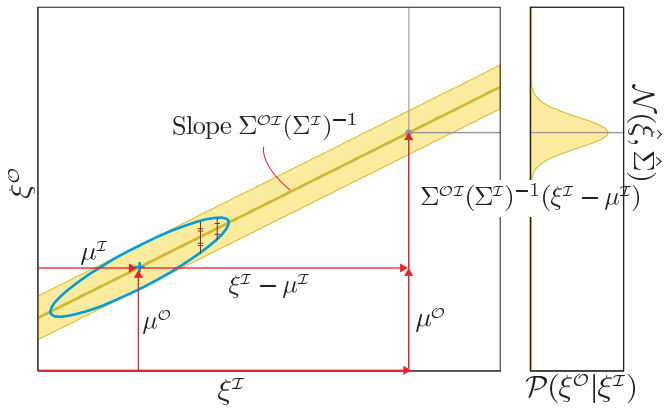

Fig. 3. The geometric illustration of Gaussian Mixture Regression inference (see also Table II). GMR approximates our dynamical systems through a non-linear weighted sum of local linear models: each regression matrix $A_k = \Sigma_k^{\mathcal{O}\mathcal{I}}(\Sigma_k^{\mathcal{I}})^{-1}$ defines coefficients of the local linear fit. Here, we display the effect of fitting with a single Gaussian a pair of input and output signals $\xi_i^O \in R^M$, $\xi_i^I \in R^P$ respectively. The projection of the regression signal to the subspace spanned by $\{\xi_i^{O,m}, \xi_i^{I,p}\}$ is a line with a slope given by the elements A_k^{mp} of the regression matrix (i.e., $\xi_i^{O,m} = A_k^{mp} \xi_i^{I,p}$). The mixture of covariance matrix in GMM defines a probabilistic envelope around the regression signal. Thus, to each input ξ_i^I is associated a probability distribution function for output $\mathcal{P}(\xi_i^O|\xi_i^I)$, with mean ξ_i^O. In the present work, we exploit the envelope to determine the boundaries for our generalized inverse kinematics solution when the solution is not exactly the regression signal ξ_i^O.

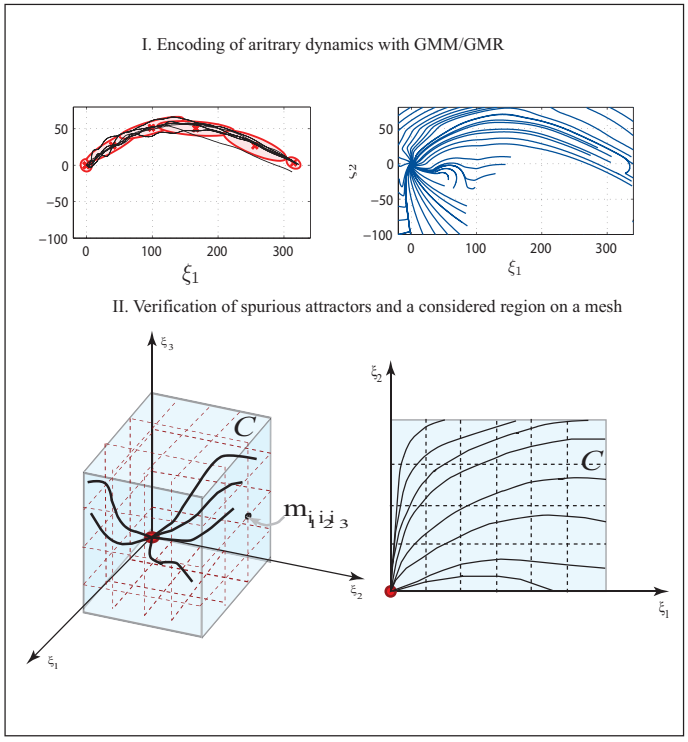

Fig. 4. I. Illustration of a GMM/GMR encoding of an arbitrary dynamics. *Top left*: Two-dimensional projection of the data with superimposed the Gaussian Mixture envelope. Top right: All trajectories regenerated using Gaussian mixture regression when starting from 20 different locations in space converge correctly to the the origin, the attractor of the system. *Bottom left and right*: in blue, the region of applicability C that embeds all demonstrated trajectories. To empirically determine if C is a region of attraction, C is sampled equally and one measures if all trajectories originating from each of sampled point converges correctly to the target.

C. Stability Analysis

Stability analysis of *linear* dynamical systems is well-studied subject [Khalil, 1996]: one either constructs a Lyapunov function for the system or analyzes the eigenvalues of the control matrix.

In contrast, there is no unique method to analyze the stability of *non-linear* dynamical systems and theoretical solutions exist only for particular cases. Classically, stability analysis of non-linear dynamical systems is performed in two steps: first, the system is linearized in a neighborhood around the points of interest (the attractors) and their asymptotic stability is verified; second, analysis of the region around the attractors is done to determine the extent of the region of attraction.

Methods to analytically estimate the regions of attraction (see Definition 7, Table I) are often based on the construction of a Lyapunov function gradually expanding its region of validity [Bai et al., 2007, Giesl, 2008, Genesio et al., 1985]. Such a procedure however produces a rather coarse estimation of the region of attraction and may fail to identify regions with non-convex boundaries. Alternative approaches take a geometrical perspective by reversing the flow of motion (by analyzing a dynamical function with a opposite sign) starting from the attractor and finding repellers and boundaries for a region of attraction from the reversed trajectories [Loccufier and Noldus, 2000]. These methods are more accurate but require considerable computation time, a known structure of an attractor's landscape (number of existing attractors and repellers).

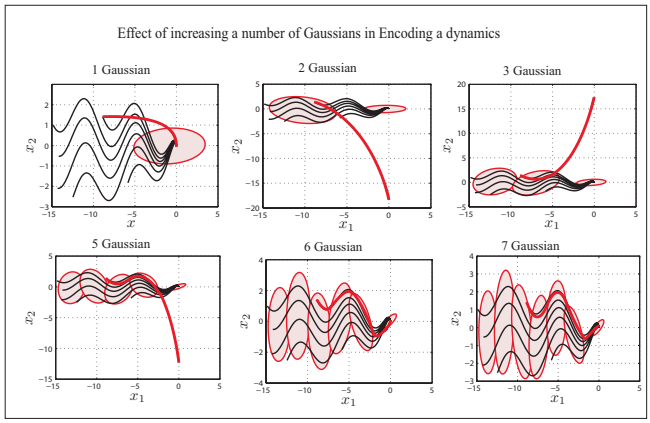

Fig. 5. Improvement in the stability of approximation with the increase the number of Gaussian components

Theoretical estimation of the region of attraction in the general case of multivariate non-linear systems is thus still an open problem. In practice, one relies on numerical procedures for evaluating whether a given region of applicability is a region of attraction. Here, we follow such an approach.

We start from the observation that GMR gives us a non-linear weighted sum of linear dynamical systems; see Eq. 9. Stability of the system is governed by the GMR parameters (the matrices A_k, B_k and mixing coefficients h_k), which are learned during training. Since the stability of the learned dynamics depends on the parameters of the training algorithm (Expectation Maximization) in Section III-D) we will show that a modification of the GMM procedure to build the mixture results in an estimate locally stable around the target.

1) Local stability at the origin: Following from the hypothesis that the origin is an attractor of the true control law $\dot{\xi} = f(\xi(t))$, we must ensure that its estimate given by (9) is also stable at the origin. Recall that for a point to be an attractor of the system (see Definition 5, Table I), there must exist a region around it where all trajectories are asymptotically stable.

Let us assume that in the neighborhood of the origin the system is governed solely by the last Kth gaussian [7]. In other words, let us assume there exists a neighborhood of the origin, where for points ξ in this neighborhood all mixing coefficients expect the Kth are zeros: $\exists B(\epsilon)$such that $\forall \xi \in B(\epsilon)\ h_k(\xi) \simeq 0\ k = 1..K-1$, where $B(\epsilon)$ is a hypersphere of radius ϵ. In this region, the system governed by Eq.9 reduces then to:

$$\dot{\xi} = A\xi + B \tag{10}$$

with $A = \Sigma_{K,\dot{\xi}\xi}\Sigma_{K,\xi}^{-1}$ and $B = \mu_{K,\dot{\xi}} - A\mu_{K,\xi}$.

The system above, driven by Eq. 10, will be asymptotically stable if the eigenvalues of the matrix A are all strictly negative. For a $m \times m$-dimensional matrix to be negative definite, all its i-th order leading principal minors should be

[7] In practice, as we seek to avoid the over-fitting, the Gaussians are sufficiently set apart, therefore at the origin the influence of all other Gaussians except for the last one becomes numerically zero.

negative if i is odd and positive if i is even; stability, therefore, is guaranteed when the following set of constraints is satisfied:

$$\|A_{\dot{\xi}\xi,[1:i,1:i]}\|(-1)^i < 0 \quad \forall i = 1, ..., m \text{ that is satisfied if} \tag{11}$$

$$(1)\, a_{ii} < 0 \text{ and } (2)\, a_{ij} \ll a_{ii} \;\forall\; i, j = 1, ..., m \text{ and } i \neq j, \tag{12}$$

where $A_{\dot{\xi}\xi} = \{a_{ij}\}_{i,j=1}^{N}$.

Figure 6 illustrates geometrically the effect of the local stability condition on the dynamics of motion and the form of the Gaussian. When projected on the $\{\xi_i, \dot{\xi}_i\}$ axes, the Gaussian corresponds to an ellipse with the main axis forming a negative slope. This results in a homogenous flow of motion toward the attractor along all dimensions.

For EM to result in such an elongated Gaussian, training data must homogeneously cover the space of motion around the target. This means that one should show the robot how to approach the target by uniformly starting all around the target. In practice, because the training set is finite and gives only a partial coverage of the state space, GMM estimate will be imprecise, resulting in both a shift of the slope of the Gaussian and a shift of the attractor's location, see Figure 6. Additional measures should, thus, be taken to guarantee the convergence to the target, which we describe next.

D. Practical approach to ensuring and analyzing stability

1) Ensuring local stability empirically: To overcome the lack of uniformly distributed training data around the origin in the experiments presented here, we generate additional so-called *synthetic* data by rotating a subset of training data, selected within a small neighborhood, around the origin. In addition, we set the center of the last Gaussian of the GMM at the target, i.e. at the origin ($\mu_{K,\xi} = \mu_{K,\dot{\xi}} = 0$), and do not update this center during training. This procedure is illustrated in Figure 6.

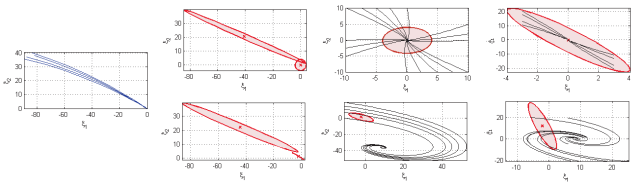

Fig. 6. Influence of the accurate positioning of the last Gaussian at the origin. *Top*: the last gaussian is positioned at the origin through addition of synthetic datapoints, that guarantees asymptotic stability of the system in the neighborhood of the origin, as can be seen from the vector field trajectories (the very right graph). *Bottom*: however, the real data asymptotically converge to the origin (the very left graph), the statistical EM does not automatically position the last Gaussian at the origin, that leads to the convergence to the spurious attractor (the very right graph).

The system driven by the truncated dynamics is given by Eq.(10) and a system generated through this procedure is ensured to be asymptotically stable within a neighborhood around the origin. Next, we describe a procedure by which we can empirically estimate boundaries of the region of applicability C.

TABLE III
MODEL TRAINING

1	Collect a dataset of demonstrations and initialize C (see Section III-D2).
2	Add synthetic data around the target
3	Choose an initial number of GMM components K ($K = 2$ in the experiments reported here)
4	**LOOP** until stable approximation is found
5	Train the joint probability $\mathcal{P}(\xi, \dot{\xi})$ with Expectation Maximization [Dempster et al., 1977]:
6	Verify local stability at the origin Eq. (12)
6	**IF** (the origin is not asymptotically stable) **THEN** increase the number of GMM components $K = K + 1$
	ELSEIF (estimate of C does not include all training trajectories) **OR** ($\exists$ spurious attractors inside the region C) **THEN** add training data **AND** retrain
	END
8	**EN**

2) Determining the region of stability empirically: As mentioned in Section III-A, estimating dynamics in the whole state-space X is impractical. Instead, we will estimate stability locally within a subset $C \subset X$. C includes training data points and lies inside the robot's workspace. Initialization of C is data-driven: size of C along each dimension is defined by the amplitude of the training dataset along this dimension.

After training, the initial guess regarding C should be restimated, to empirically verify that C is a region of attraction of the origin and that it does not include any other attractors . We follow a numerical procedure in which we integrate trajectories forward starting from a uniform mesh defined on the boundaries, and verify that all the trajectories converge toward the origin.

To do this, we construct a mesh M covering boundaries of C: $M(\tau_1..\tau_N) = \{(\xi_{i_1}^1..\xi_{i_N}^N) = (i_1\tau_1..i_N\tau_N),\ i_1 = 1..n_1, ...,\ i_N = 1..n_N\}$, where $\tau_1 = c_1/n_1\ ..\tau_N = c_N/n_N$, $c_1\ ..\ c_N$ – size of each of dimensions of C; $n_1..\ n_N$ – size of the mesh along each of dimensions in $\mathbb{R}^N$ (see Figure 4-II).

We integrate trajectories starting from each node $(\xi_{i_1}^1..\xi_{i_N}^N)$ on the mesh M and verify that the velocity is zero only at the origin, thus ensuring that only the origin of the system is an attractor. If this condition is satisfied all trajectories starting inside C will not leave the boundaries, due to the properties of differential equations.

To improve stability, we increment the number of Gaussians K and re-estimate the system using EM. Augmenting the number Gaussians allows a more precise encoding of the dynamics locally along the trajectory; see Figure 5. Since instabilities result often in the motion exiting the desired trajectory (e.g. if there are sharp turns in the trajectory that have been poorly approximated by the mixture), increasing the granularity of the encoding ensures that the system will be better guided along the various non-linearities of the trajectory.

Table III summarizes the steps of the complete procedure by which we iteratively test and re-estimate the system to improve

and ensure local stability within the domain C.

IV. EXPERIMENTAL RESULTS

To validate the performance of the proposed method itself without blurring it with noise inherent to human demonstrations, we first tested its ability to reconstruct given theoretical dynamical systems. With a known system we may generate a clean training set, learn an approximation of the dynamics and further compare how well the learned dynamics approximate the real one.

Further, we verify the applicability of the method to robotics by teaching two robots manipulation tasks. We report on each of these next.

A. Learning Theoretical Dynamics

The method was validated to estimate four two-dimensional dynamical systems (*Systems 1-4*) and one three-dimensional dynamical system (*System 5*), each of them contains different number of attractors and exhibits different stability properties. In each case, we generated six trajectories using the theoretical dynamics and used these for training the GMM. When the dynamical system had more than one asymptotically stable attractor, trajectories were generated only in the subpart of the state space around one of them.

Note, the legend for Figures 7 - 10 is described in Figure 1. Each of the figures encompasses, in the first row, plots giving a general view of the original dynamics with vector fields (a) and three-dimensional phase plots (b-c), in the second row, a view of the GMM superimposed to the training data, and in the 3rd row, vector field (a) and phase plots (b-g) of the the estimated dynamics superimposed on the original dynamics.

System 1.

$$\begin{aligned} \dot{x}_1 &= -x_1 + 2x_1^2 x_2; \\ \dot{x}_2 &= -x_2. \end{aligned} \tag{13}$$

The system has a single locally asymptotically stable equilibrium point at the origin. We approximate the dynamics of this system in a region $[-4; 0] \times [0; 2]$, where it is locally asymptotically stable. Results are presented in the Figure 7.

System 2

$$\begin{aligned} \dot{x}_1 &= 700 - 2x_1 + 200x_2 e^{\frac{25x_1 - 10^4}{x_1}}; \\ \dot{x}_2 &= 1 - x_2 - x_2 e^{\frac{25x_1 - 10^4}{x_1}}; \end{aligned} \tag{14}$$

The system has two equilibrium points – one asymptotically stable ($x_1 = 335$; $x_2 = 0.089$) and one unstable ($x_1 = 489$; $x_2 = 0.5$). We approximate the dynamics in the region $[0; 400] \times [-2; 2]$, where it is locally asymptotically stable. Results are presented in Figure 8.

System 3

$$\begin{aligned} \dot{x}_1 &= -x_2; \\ \dot{x}_2 &= x_1 - x_1^3 - x_2; \end{aligned} \tag{15}$$

The system has three equilibrium points - two unstable ($x_1 = -1; x_2 = 0$ and $x_1 = 1; x_2 = 0$) and one asymptotically

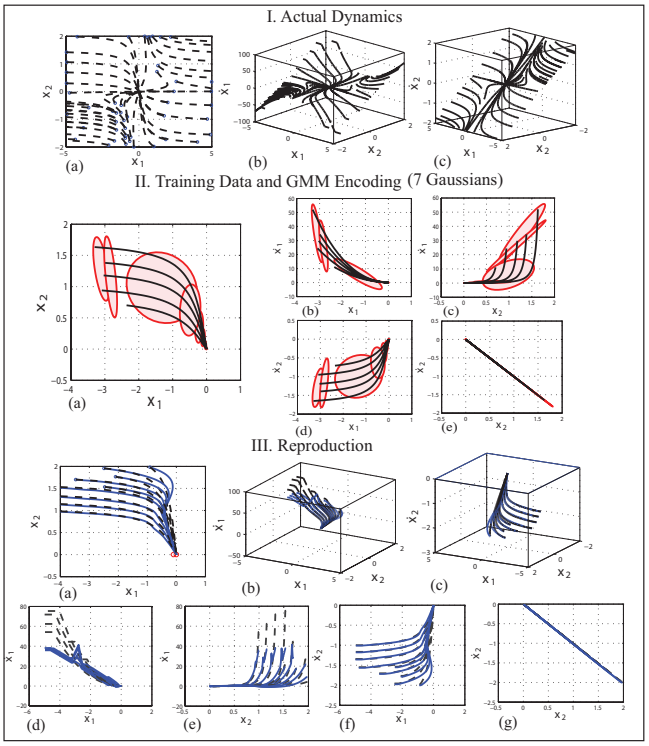

Fig. 7. *System 1.* The proposed method encodes this system with 7 Gaussians; the learned system exhibits good precision in the area covered by demonstrations, outside this area the precision is also admissible except for a region in the direct proximity to y-axis, where actual trajectories represent an excess curvature as approaching to the equilibrium, e.g., a trajectory starting at the bound $x_2 = 2$. In this region, a flat part of trajectories is reproduced well, though the steep parts that were not demonstrated are attracted towards the region covered by the training set.

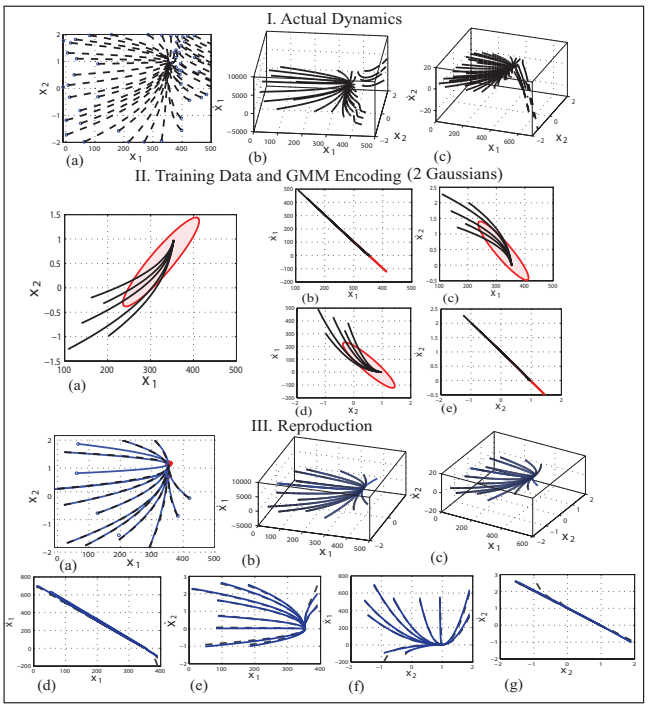

Fig. 8. *System 2.* As the behavior of the system in the considered area is relatively simple, 2 Gaussians are sufficient to achieve the good performance, even in areas unseen during demonstration. Interestingly, the learned dynamics is extrapolated very well beyond the area covered by the training set.

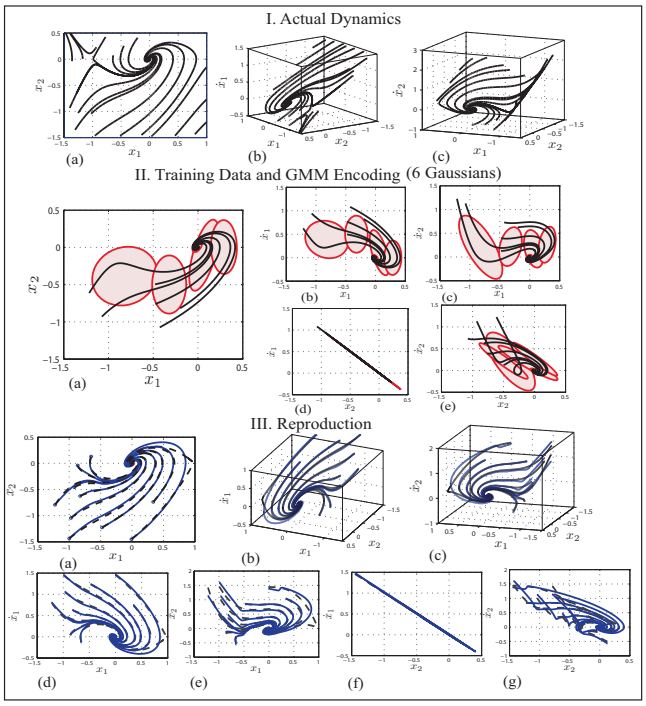

Fig. 9. *System 3*. Despite manifest non-linearity in the trajectories, the dynamics is successfully approximated with 6 Gaussians. Note, even unseen, circular shape trajectories (starting around $x_2 \approx 0$) are reproduced correctly in both position and velocities spaces.

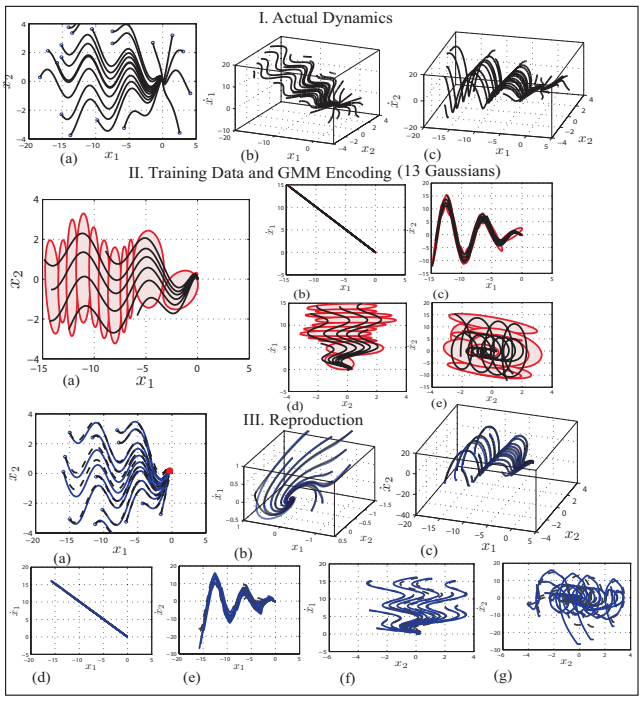

Fig. 10. *System 4*. The system is strongly non-linear, 13 Gaussians are necessary to achieve a good precision in the considered region. Complex dynamics and increased number of Gaussians lead to less strong generalization abilities of the method. Indeed, trajectories started beyond the region covered by the training set tend to depart from the real trajectories generated by the dynamics, it is particularly noticeable in the velocity space, see section III-(g). However, even in this non-trivial case the system generate admissibly good results from few demonstrations.

stable $x_1 = 0; x_2 = 0$. We approximate the dynamics of this system in a region $[-1.5; 1] \times [-1.5; 0.5]$, where it is locally asymptotically stable. Results are presented in Figure 9.

System 4

$$\begin{aligned} \dot{x}_1 &= -x_1; \\ \dot{x}_2 &= -x_1 \cos x_1 - x_2; \end{aligned} \tag{16}$$

The system exhibits strong nonlinearity due to the cosine term; the system is globally asymptotically stable and converges asymptotically to the origin. We approximate the dynamics of this system in a region $[-20; 0] \times [-4; 4]$. Results are presented in Figure 10.

System 5

$$\begin{aligned} \dot{x}_1 &= -x_1 - x_2 + x_3^2; \\ \dot{x}_2 &= x_1 + 10 \cos x_2 * x_2 - x_3^2; \\ \dot{x}_3 &= x_1 + 2x_2 - x_3; \end{aligned} \tag{17}$$

Locally asymptotically stable three-dimensional dynamics

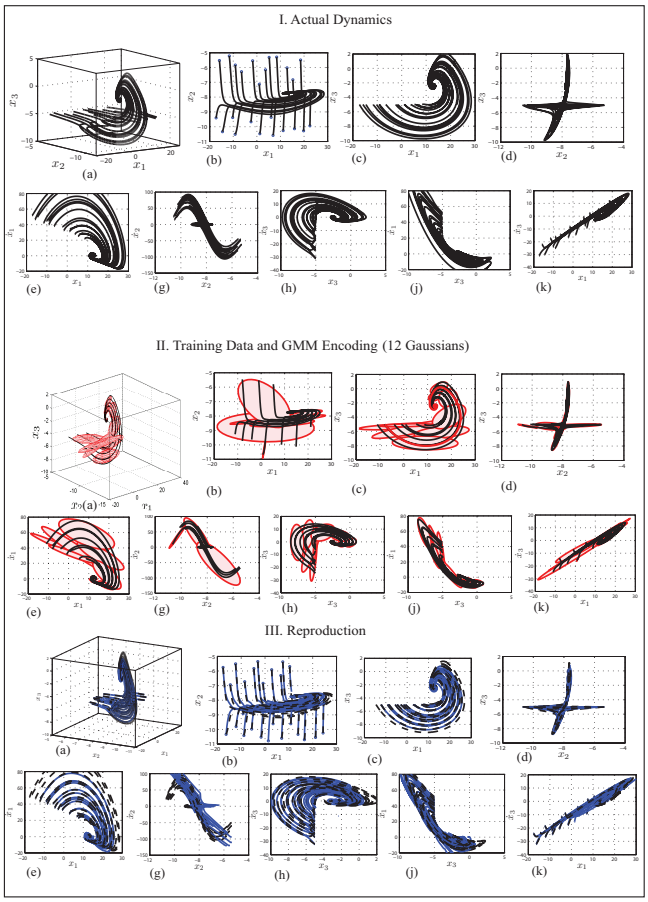

Fig. 11. *System 5*. Strongly non-linear 3D dynamics. In this case, a slight increase in a number of demonstrations allows for accurate approximation and generalization.

with a single attractor at $[12.98; -7.75; -2.5213]$. We approximate the dynamics of this system in a region $[-20; 30] \times [-11; -5] \times [-10; 2]$. Results of the learning process are presented in the Figure 11.

1) Quantification and discussion of results: Quantification of results achieved on both theoretical systems and actual robotic motions are presented in Table IV. As it can be seen all systems permit coarse representation with a relatively small number of Gaussians, moreover such a sparse representation achieves good precision in reproducing the actual dynamics. Furthermore, as shown in Figures 7-11, the system can generalize outside the training domain. This property is particularly useful for practical applications as this allows to predict the behavior of the system outside the region covered during training, hence reducing the amount of training data required. In the examples covered here, only 6 training trajectories were required in each case.

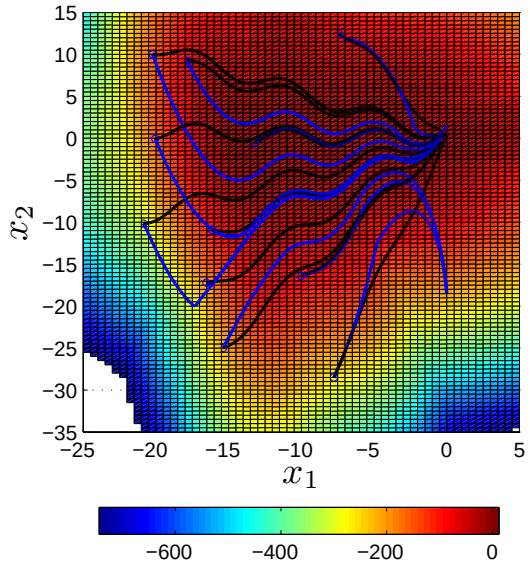

Fig. 12. Extrapolation properties of the GMMs encoding. A color map reflects changes in values of the the likelihood (18) of datapoints, the dark-red area represents an area of the most reliable inference regarding the velocity. For reconstructed trajectories starting outside this area, the deviation from the actual dynamics may be considerable. Interestingly, in the region of attraction of the origin, trajectories are strongly attracted towards a region covered by the training set. It is a useful property for practical applications as this allows to predict the behavior of the system outside the region covered during training, hence reducing the amount of training data required.

Note that, since the dynamics is learned from data covering only a subpart of the domain, it does not necessarily have the same attractor landscape and the region of attraction across the complete domain as the original system, even if it accurately approximates the original system locally. For example, in System 3, the original dynamical system has three equilibrium points, while its approximation has a unique asymptotically stable equilibrium. To overcome this, one may provide additional demonstrations covering dynamics in the neighborhood of the other equilibriums: Figure 15 presents results of learning the dynamics around the two different attractors of *System 5*. The demonstrations were provided in the neighborhood of the two asymptotically stable attractors; during learning, positions of two Gaussians were fixed on the attractors, and the algorithm was running to verify local asymptotical stability of both attractors. The regions of approximation C were analyzed separately for each attractors. The learned system managed to accurately grasp the complex dynamics, further, it allowed to separate the two flows of trajectories leading to different attractors based on the initial conditions of motion.

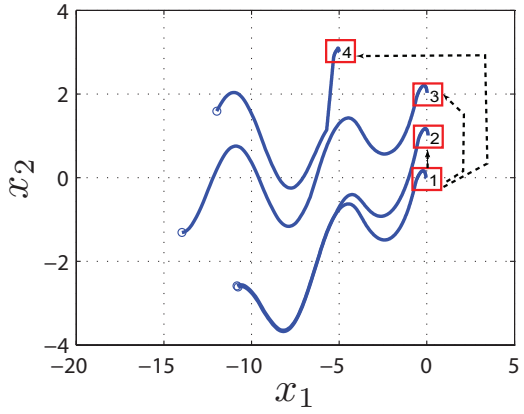

Fig. 13. Robustness to perturbations. The target is shifted several times (to positions 2, 3, 4) after the onset of motion.

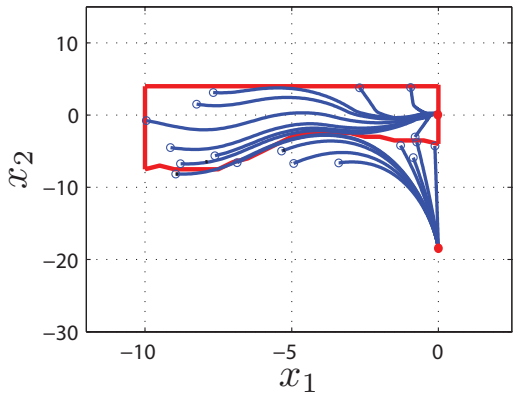

Fig. 14. Numerically estimated region of stability for the System 6; updated boundaries of the considered region taking into account boundaries of RA are plotted in red. An actual and spurious attractors are red circles. We extended a considered region from $[-20; 0] \times [-4; 4]$ to $[-10; 0] \times [-10; 4]$. Note, the numerical method estimated a lower bound that goes along a trajectory with a good precision. Other bounds were left unchanged, i.e., in the other directions the considered region does not cross boundaries of the region of attraction.

In addition to stability of reproduction, one should keep in mind that the considered region of applicability should not exceed a region where the likelihood of observing new data allows performing a confident inference regarding the velocity. In Figure 12 we depict how the likelihood changes beyond the region covered by the training set. Likelihood was computed as follows:

$$\mathbf{L}(\xi) = \log\left[\max_i h_i(\xi)\right]. \tag{18}$$

L gives a measure of the maximum probability of a point ξ to belong to any of the K Gaussians. The region where L exceeds a given threshold[8] represents the region where the system can still make a confident probabilistic inference. Note

[8]We took an empirically chosen threshold of -10.

TABLE IV
QUANTIFICATION OF RESULTS

System	Dimension	Number of demonstrations	Number of GMMs components	Number of iterations in model training[1]	Precision [2]
System 1	2	6	7	252	$8 \cdot 10^{-3}$
System 2	2	6	2	70	$3 \cdot 10^{-3}$
System 3	2	6	6	150	10^{-4}
System 4	2	6	13	475	10^{-1}
System 5	3	10	12	393	$8 \cdot 10^{-2}$
KATANA experiment	3	4	4	132	-[3]
HOAP experiment	3	4	5	160	-[3]

[1] The algorithm iterates until the change in the likelihood falls below 10^{-8}

[2] Precision is computed as a mean square error, on both seen and unseen trajectories, according to: $\frac{\sum_{i=1}^{M} \|\hat{\xi}^i - \xi^i\|^2}{M \cdot \Delta\xi}$, where $\hat{\xi}^i$ – learned trajectory ξ^i – theoretical trajectory, $\Delta\xi$ is an average amplitude of motion.

[3] Estimation of precision is non-applicable due to the presence of noise in the training data.

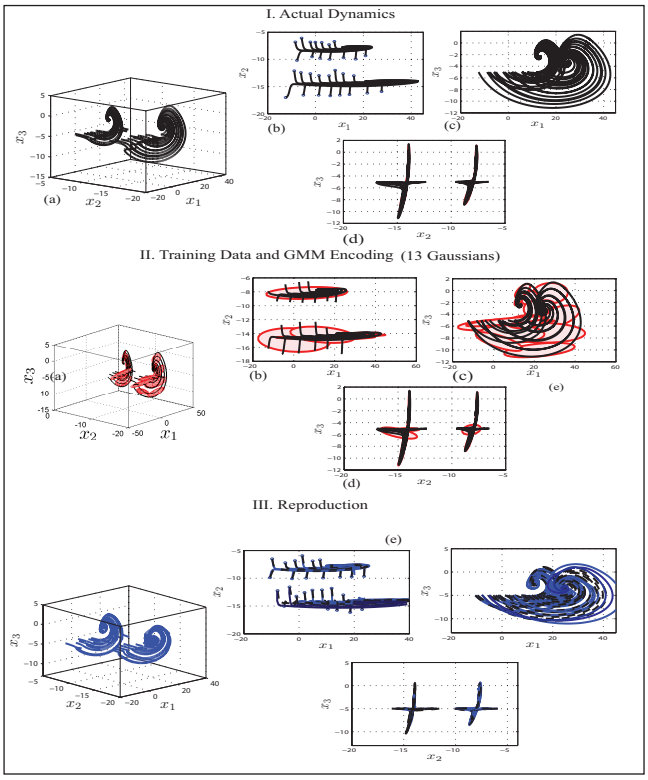

Fig. 15. Learning motion with two attractors. 3-dimensional trajectories are generated by *System 5* that displays a periodic behavior. Trajectories were demonstrated in the neighborhood of two asymptotically stable attractors. During the reproduction, the system managed to accurately reproduce dynamics around both attractors.

that all the trajectories that start in areas where L is too small will significantly depart from the real dynamics. This is due to the effect of the weights h_i associated to each Gaussian and how these influence the direction of the velocity vector: nearby the demonstrations, the influence of the closest Gaussian dominates that of all Gaussians, hence guiding closely the motion. However, far away from the demonstrations, the influence of all Gaussians becomes comparable and the resulting direction of velocity may point away from the signal.

As mentioned in the introduction, an inherent property of stable dynamical systems is their robustness to spatial and temporal perturbations. Figure 13 illustrates this aspect for one of the learned dynamical system, when the target is moved after the onset of the motion. As we see, the trajectories adapt smoothly to the change. Note, however, that the velocity profile may change abruptly when the perturbation occurs. To overcome this drawback it would be necessary to consider second-order dynamics.

As discussed previously, the GMMs encoding may result in spurious attractors outside the empirical stability domain C and in regions with low likelihood, see, e.g., Figure 12.

There are several reasons for the emergence of spurious attractors: first, the training set gives only a partial and noisy representation of the dynamics. Providing additional data in the regions around spurious attractors usually improves greatly performance. Second, the shape of the signal influence greatly stability. For instance, if the curvature of the trajectories changes smoothly, the spurious attractors, if any, will usually lie outside of the region of the confident inference, see Figure 12. However, if the system trajectories experience sharp changes in the curvature, as e.g., System 1, see the Figure 7, the likelihood of having spurious attractors in the considered region increases. By adding more Gaussians around the point with a sharp curvature one increases the guidance provided by the GMM and thus decreases the chances. By considering these practical shortcomings, one may improve a particular encoding to achieve the admissible performance.

V. APPLICATION TO ROBOT CONTROL

Further, we validate the method to learn the dynamics of motion of a robot endeffector when trained through human guidance. Here, the dynamics of motion becomes the control law that iteratively moves the robot's arm along a trajectory.

A. Encoding motion in the operational space

Since the framework we defined above does not make any assumption as to the type of variables to be used for training, we are unconstrained in our choice of variables for controlling a robot. Here, we choose to describe motions according to the following variables: the translation component of motion of the end-effector is described by a vector of Cartesian coordinates $\mathbf{x} \in \mathbb{R}^3$.

Each demonstrated trajectory is, thus, represented by the following dataset: $D = \{\mathbf{x}_t, \dot{\mathbf{x}}_t\}_{t=1}^{M}$, where M is the number of datapoints in a trajectory. To reproduce a task, we first learn an estimate of the dynamical system using the method described

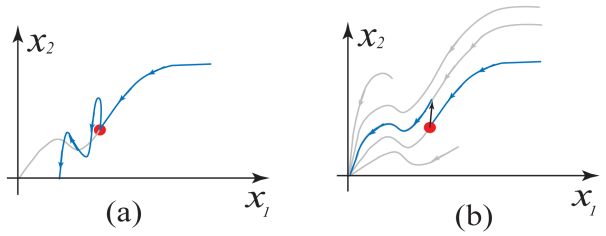

Fig. 16. (a) If a trajectory in the operation space passes through non-reachable joint positions IK may return velocity in the operation space that sends a robot too far from original trajectory, so linear assumptions of approximation of kinematics does not satisfy and overall trajectory tracking will fail. (b) In the case of motion encoding with a dynamical system, after perturbation the robot will not try to return to the previous trajectory violating the linear approximation of kinematics, instead the dynamical system will generate other trajectory from the point where the robot occurs.

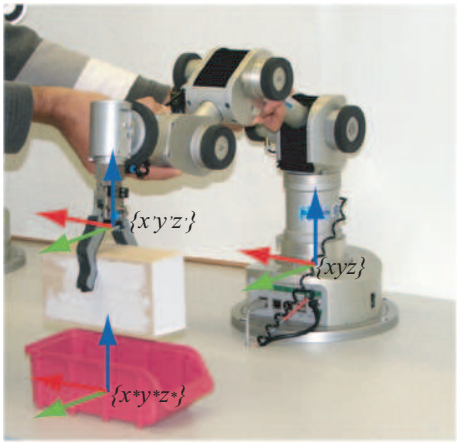

Fig. 17. We encode tasks in a referential located at the target and moving with it$\{x^*y^*z^*\}$; this referential is expressed in the fixed global referential $\{xyz\}$(usually we choose one attached to static parts of a robot). Actually, the motion of the robot end-effector is expressed as moving a referential associated with the end-effector $\{x'y'z'\}$.

in Section III-A and then use the Moore-Penrouse pseudo-inverse to compute the corresponding joint angles. Table V summarizes the steps of the reproduction algorithm.

B. Set-up

We validated the above method in two practical tasks, see Figure 18 where a human teacher guides the robot through the motion. We also implemented the learned 3-dimensional *System* 5, as a control law for our robot. To demonstrate the generic character of the approach we ran experiments with two different robotic platforms: a 6 degrees of freedom industrial-like KATANA arm from Neuronics and a 4 degrees of freedom robot arm of the humanoid robot HOAP-3 from Fujitsu.

C. Experiments with KATANA

The first experiment consists in the KATANA putting an object into a container. Here, the KATANA arm was taught to put a rectangular wooden brick into a rectangular container; see fig.18-left.

TABLE V
ON-LINE TASK REPRODUCTION

1	Assume that a controller $\hat{f}_{\mathbf{x}}$ has been learned, the robot is thus ready to reproduce a task
2	Detect a target position in the global referential $\{xyz\}$; see Figure 17: $\mathbf{x}^*$
3	Recompute the current position of an end-effector in the target referential $\{x^*y^*z^*\}$: $\mathbf{x}_0$
4	**LOOP** until the target position is reached
6	infer the velocity for the next iteration t through GMR Eq.9: $\dot{\hat{\mathbf{x}}}_t$ $\dot{\hat{\mathbf{x}}}_t = \sum_{k=1}^{K} h_{k,\mathbf{x}}(\mu_{k,\dot{\mathbf{x}}} + \Sigma_{k,\dot{\mathbf{x}}\mathbf{x}}\Sigma_{k,\mathbf{x}}^{-1}(\mathbf{x} - \mu_{k,\mathbf{x}}))$
8	Solve the Inverse Kinematics problem to find: $\dot{\mathbf{x}}_t, \dot{\boldsymbol{\theta}}_t$
9	compute a new position $\mathbf{x}_t, \boldsymbol{\theta}_t$
10	**END**

In the second experiment, the KATANA was controlled with *System 5* with the origin of the system positioned on an arbitrary object. This experiment meant to test the ability of the learned system to generalize to context unseen during training and to quickly adapt to perturbations.

D. Experiments with HOAP-3

The clench of the HOAP-3 is rather small, therefore it can grasp only thin objects. In this task the robot had to grasp a box which is thin along one dimension, so the robot should follow a specific path to properly position its hand; see fig.18-right.

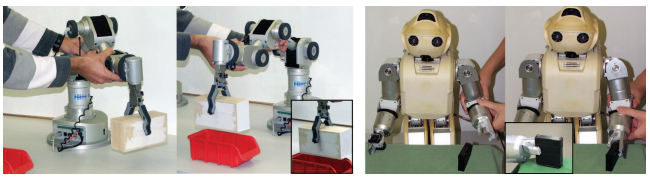

Fig. 18. Set-up of the experiments. Left: KATANA puts a wooden brick into the container, to achieve the task the robot should lift the brick and move it following an elevating trajectory. Right: HOAP-3 grasps a box, to achieve the task HOAP should approach the box with a specific orientation and than lower its arm, as the clench is small, see small figure in the corner.

During training, the robots were shown the tasks 5 times by a human user guiding their arms. Values of the robots joints were recorded during this passive motion and used for reconstructing the position of the end-effector.

E. Results of Learning Dynamics from Motion Data

After training, we tested the system by requesting the robots to reproduce the tasks in various conditions. The results of the experiments are summarized in Figures 19-22.

To test the generalization abilities and the stability to perturbations we performed experiments in different conditions, by changing the starting positions of the robots and shifting the container (for the KATANA's experiment) or the box (for the HOAP-3's experiment). Results are presented in Figure 22; in both experiments learning of position control was

successful and the robots all reached successfully the targets and accomplished the tasks.

Results of generalization for the second experiment with KATANA reproducing *System 5* are presented in Figure 20 - II. The area where demonstrations were provided is depicted in Figure 20 - II (b) with red squares. The system further allowed to reproduced the motion starting from any position of the sub-space of the workspace, depicted in grey. Note, that even few demonstrations provide good generalization properties.

The ability to generate a trajectory from arbitrary initial position to the target with a relevant velocity profile is a strong point of encoding motion with Dynamical Systems in the state-space, furthermore it provides real-time adaptation to perturbations in the position of the target. The Figure 20-I presents results of tracking a marked object mapped into the attractor of the dynamical system. After shifts of the target, the robot finally reaches the object following the demonstrated position and velocity profile.

VI. DISCUSSION AND FUTURE WORK

Below, we discuss the major hypotheses postulated in this work together with possible alternative solutions.

A. *Multi-dimensional systems, first order dynamics*

The method proposed here allows learning of non-linear multivariate dynamics where the correlation between the variables is important. Other works on dynamical control consider each degree of freedom separately, hence discarding information pertaining to correlation across the joints. While storing correlations across the joints is costly (in GMM, it forces one to compute the complete covariance matrix, rather than computing only the diagonal elements), it is advantageous as correlations contain features characteristic of the motion. For instance, in bimanual coordination tasks in which left and right arms should follow different dynamics while doing so in coordination [Gribovskaya and Billard, 2008], embedding the correlations in the representation ensures the reproduction of both the dynamics of each arm and the correlations across the arms. Furthermore, learning correlation between a multivariate signal and its derivatives allows to considerably decrease a number of Gaussians required to accurately encode the training dataset.

While we started with the hypothesis that the control law followed a first order dynamics, the method proposed here may be extended to learn higher-order dynamics (as higher-order systems can always be expressed in the canonical form as a set of first-order systems). That is particularly relevant for applications where it is necessary to control the acceleration profile. We intend to address this problem in future work.

Potential difficulties concerning shifting into higher-order derivatives that can be envisioned, are associated with the increased dimensionality of a resultant statistical problem. With an increase in the number of dimensions, a stable approximation would require more training data or need to introduce certain heuristics to partially decouple the problem into a set of systems with lower dimensions.

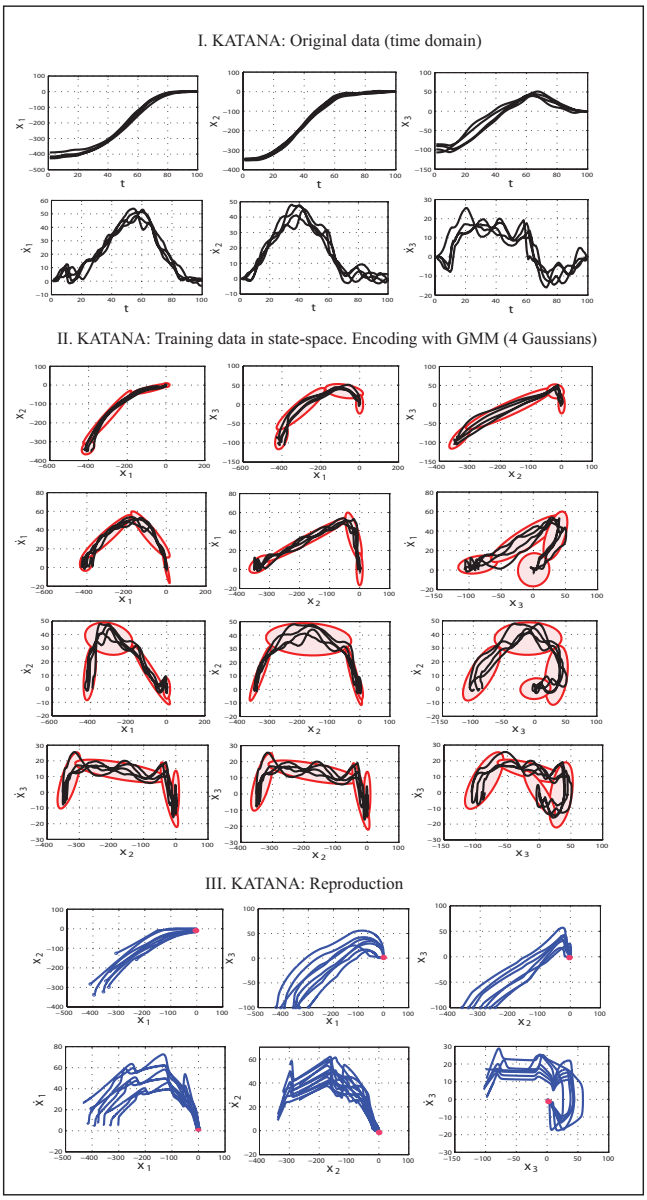

Fig. 19. KATANA experiment 1: Results of encoding and reproduction of the experiment where KATANA had to put a brick into a container.

B. *Time independency* vs *time dependency*

In this paper, we advocate that time-independent encoding in the state-space offers more robust representation in comparison to traditional time-dependent encoding. Results confirmed that for a certain range of motions, the state-space representation is indeed highly robust to spatial and temporal perturbations. Moreover, it allows to reproduce tasks even in unseen parts of the workspace.

Yet, certain motions, such as those requiring the synchronization with an external dynamics, should be encoded using a time-dependent representation or, if the external dynamics is known, using an explicit parametrical coupling of two time-independent dynamics, such as that done in [Ijspeert et al., 2001]. Another limitation of the time-independent representation relates to the possibility of encoding compound motions: in this case, the whole motion may be segmented into a set

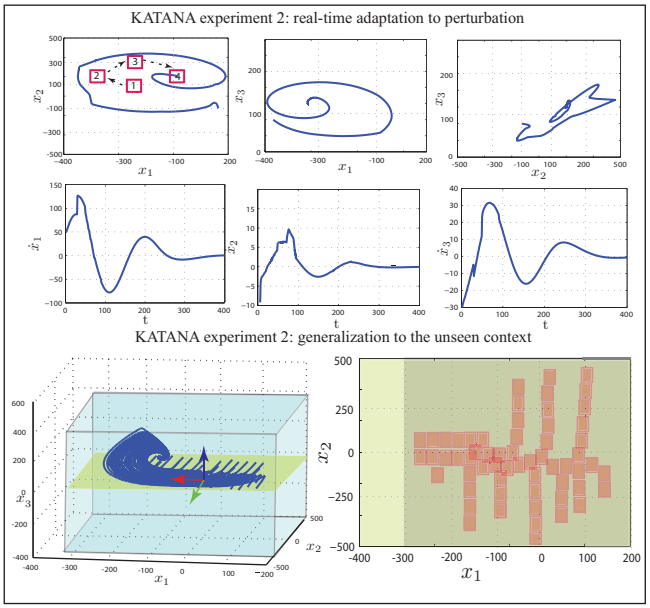

Fig. 20. KATANA experiment 2: *I. Real-time adaptation to perturbations*. The target was consequently shifted from the position 1 to the position 4. First row: trajectory of the robot's end-effector; second row: velocity profile. *II. Generalization to the unseen context*. (a) The workspace of KATANA is highlighted by the blue box, the reproduction was systematically tested starting robot from positions on the yellow plane. (b) The starting plane from the robot workspace is in yellow. The robot was required to reproduce the motion from points monotonically covering the part of the starting plane (in grey). For comparison, the part of space, where the demonstrations were provided is in pink. Notice, that demonstrations are sparse, but the system manages to generalize to other parts of the workspace.

of simpler ones governed by a single attractor. However, the problem of how to transit across these systems remains an open issue.

C. Kinematic controller

In the experiments reported here, control of the robot was purely kinematical, encoding the desired kinematic trajectories, but not taking into consideration the dynamical properties (actual torques) of the robot limbs. An additional control step was then necessary to convert positions into motor commands by means of the inverse dynamics (KATANA) or a PID controller (HOAP-3). Learning the inverse dynamics, while a highly value topic in itself, is beyond the scope of the present paper. Further, considering that many of the current robotic platforms are position-controlled, while providing position feedback in real-time, the proposed approach is thus valid for a large set of applications.

D. Choice of statistical framework

GMMs being a global statistical techniques (by opposition to local non-parametric methods such as LWPR, GPR) was shown to be suitable for estimating dynamics from sparse demonstrations, that are typical of programming by demonstration applications. However, neither GMMs nor LWPR and GPR ensure stability of a learned approximation. Here, we proposed an algorithm that leads to local asymptotical stability and gradually improves the quality of the approximation while

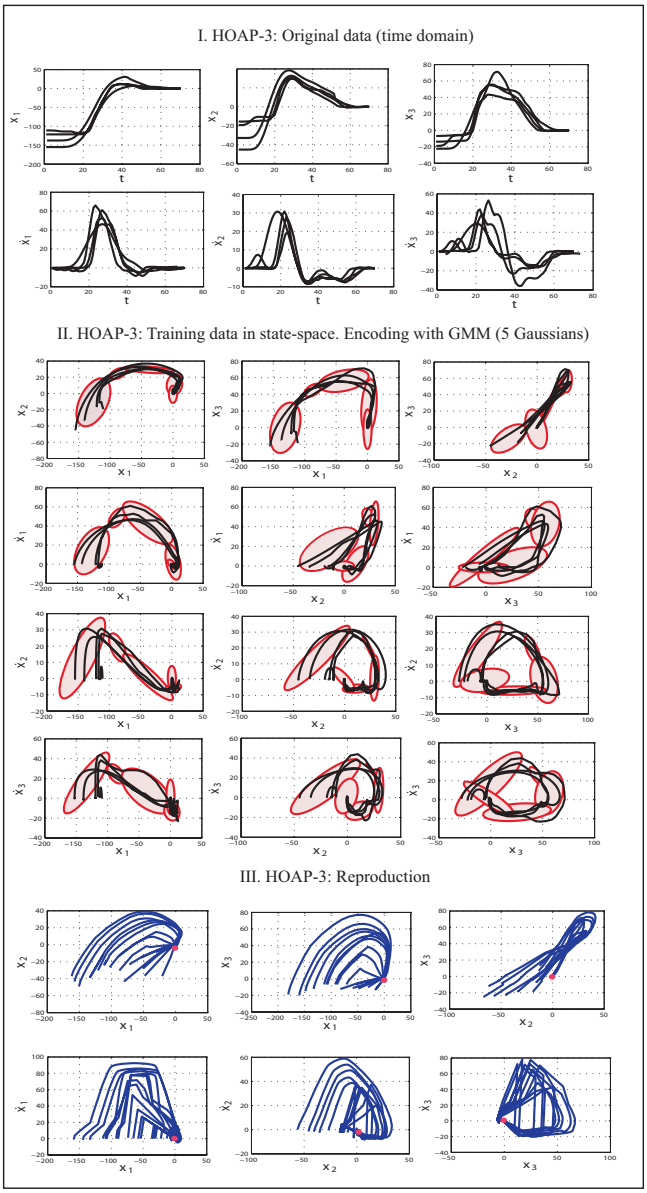

Fig. 21. HOAP-3 experiment: Results of encoding and reproduction of the experiment where HOAP-3 had to grasp a box.

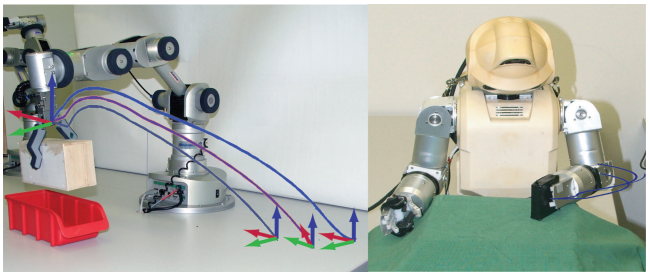

Fig. 22. The results of reproduction of dynamically generated trajectories on the robots. To check the generalization abilities of the learned dynamics the trajectories were reproduced from different initial positions.

widening the region of applicability C. Potentially, the same procedure may be adopted for other statistical frameworks. However, the accuracy of the approximation may significantly vary depending on a particular choice.

One should note that EM is more computationally expensive than LWPR with a number of iteration steps during training of $O(K \cdot M \cdot N)$ in comparison to $O(N)$. Both of these however remain small in comparison to GPR. Similarly to LWPR and in contrast to the GPR-based methods, GMR's computational costs for the retrieval procedure are low and increase linearly with the number of parameters. Additionally, GMM-based models result in much less parameters due to the coarse representation.

E. Real-time adaptation to perturbation vs traditional planners

One of the strengths of the proposed approach is its ability to cope with perturbations in real-time. By perturbation we referred to unexpected changes in the positions of the attractor or of the robot's joints during motion. We demonstrated how the learned dynamics with a position of an object mapped into an attractor can successfully track the object. Such a flexibility combined with the guarantee of ultimately reaching the object is one of the major advantages of the proposed method in comparison with traditional planners [Yokoi et al., 2009, Yoshida et al., 2008, Diankov and James Kuffner, 2007, Kuffner et al., 2002]. One should emphasize that planners, in turn, are advantageous when the environment is known and for providing mechanisms for obstacle avoidance. The latter is, however, achieved by introducing a heuristic-based cost function that penalizes certain directions. Potentially, our approach may be combined with such a cost function that perturb an output of a learned dynamical system pushing it away from obstacles.

Note, that our system though introduces certain hypotheses, still remains rather generic regarding tasks it may reproduce, furthermore, it may work with limited and inaccurate information about the environment, as it does not require any costly replanning. At the same time, to benefit from optimal planning and capacity for obstacle avoidance, one should provide an algorithm with precise information regarding objects in the workspace and introduce certain task-related heuristics to improve convergence.

F. Single vs *several attractors*

A further hypothesis pertaining to the work presented here was the idea that the dynamical system to be discovered had a single or several known fixed point attractors. This can be considered as a limitation, as a dynamics may be governed by the existence of more complex orbits than merely fixed points. For example, an arbitrary free motion may have a particular curve in space as attractor. The applicability of the proposed method in this case will mostly depend on the quality of training data; further no stability can be guarantee. Procedures for ensuring stability of complex orbits may substantially widen the class of motion under consideration, covering dancing or sport motions that are usually characterized by the existence of certain curves to which all trajectories converge.

G. Training data

The generalization properties of dynamical controllers directly depend on the quality of training data; the aspect common to all statistical learning methods. It might be compensated in different ways: 1) by providing an exhaustive set of accurate demonstrations; 2) by allowing a robot to explore on its own (considered in Reinforcement Learning [Guenter et al., 2007]); 3) by providing more variability in a limited set of demonstrations (the problem has been discussed in [Calinon and Billard, 2007]). The first option does not agree with a requirement of user-friendliness of teaching interfaces, as a number of demonstrations should be kept bearable for a user; the second approach may require additional time; therefore, we concentrate on improving quality of demonstrations by introducing more variability into a small set of demonstrations.

H. Kinesthetic teaching

For demonstrating tasks we used the kinesthetical teaching approach that consists of directly demonstrating the task using a robot's own body. One of advantages of this approach is that the human can feel limitations of the robot's architecture and adapt his/her intuition about an optimal or efficient motion accordingly. Although we actively exploit this learning paradigm, other approaches such as vision-based learning are also widely used and can be more intuitive for humans. Our system may be applied to the motion data obtained through different modalities.

I. Practical consideration

From a practical point of view, mapping position of manipulated objects into attractors of Dynamical Systems considerably improves the precision of motion at a target and therefore allows considering prehensile tasks in the framework of Programming by Demonstration; where so far generation of large-scale motions has been addressed.

The approach was shown to be generic in that it did not depend on the particular geometry of the robot's arm, nor on the particular variables to be learned. Indeed, it could be successfully implemented to control robot arms with different geometries and for learning the dynamics of different variables inherent to position and orientation control. **Source code and supplementary material is available at http://lasa.epfl.ch/elena/learning-dynamics.htm**

VII. Summary

In this paper, we proposed a method for learning a nonlinear multi-dimensional dynamics of motion through statistically encoding demonstrated data with Gaussian Mixtures. Further, we addressed the problem of ensuring stability of a resultant control law: first, we formulated conditions that parameters of GMMs should satisfy to guarantee local asymptotical stability of an attractor, then we proposed a numerical procedure to verify boundaries of the region of applicability where the control law can be securely applied.

To test the method, we conducted two types of experiments: 1) learning theoretical dynamics with known mathematical forms to estimate the accuracy of approximation and 2) learning dynamics of manipulation tasks recorded with two different robotic platforms to assess the applicability of the approach to the noisy data. In all experiments the system demonstrated good results in terms of high accuracy during reproduction, ability to generalize motions to unseen contexts, and ability to adapt on-the-fly to spatio-temporal perturbations. We also showed how the system can encode more than one attractor, successfully reproducing each separate dynamics locally around each attractor and separating the flows leading to the different attractors.

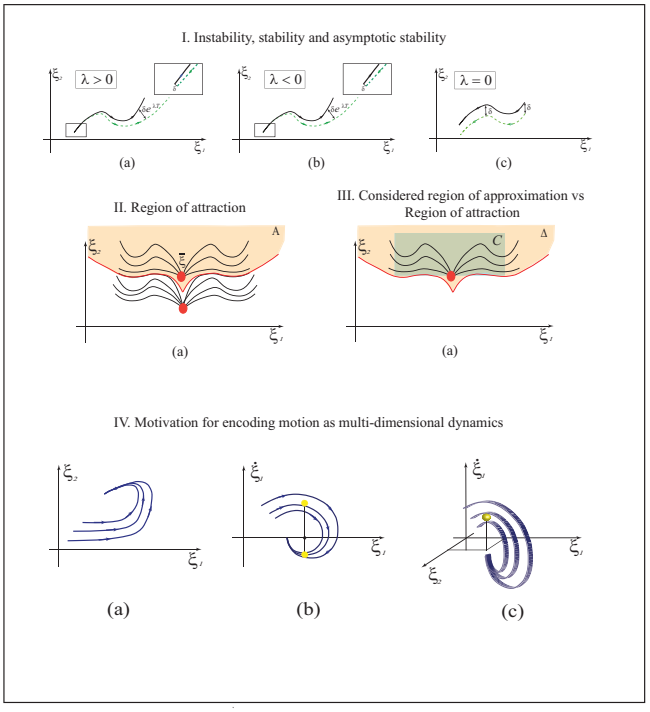

Fig. 23. Appendix II. Geometrical illustration of stability and multi-dimensional correlation in the state-space. I. ***Stability problem***: stability of a dynamical system is defined by a maximum value of its *Lyapunov exponent* λ (in the linear case, it coincides with eigenvalues of a control matrix). (a) In systems with *negative* Lyapunov exponents volume between trajectories contracts; (b) In systems with *positive* Lyapunov exponents two arbitrary near trajectories diverge from each other exponentially fast. In the linear case, one may easily find Lyapunov exponents and estimate the global behavior of the overall system. In the non-linear case, the system may have different Lyapunov exponents in different parts of the state-space, moreover, non-linearities make analytical investigation of properties particularly tedious. IV. ***Multi-dimensional dynamics*** Analyzing dynamics of vector-valued timeseries requires their encoding in multi-dimensional state-spaces. Generally, one cannot unambiguously decouple dynamics of each dimension. Consider a simple 2D motion in Figure II-(a), the phase-space of this motion in $\{\dot{x}_1, x_1\}$ is in Figure II-(b): for each value x_1 there exist two different values of velocity, therefore, it is not possible to unambiguously encode dynamics of motion as two decoupled system $\dot{x}_1 = f_1(x_1)$, $\dot{x}_2 = f_2(x_2)$. However, if one look at the dependence $\dot{x}_1 = f(x_1, x_2)$ depicted at Figure II-(c) this ambiguity can be easily eliminated. This problem is know in the literature on Dynamical Systems as a problem of searching for a *minimum embedding dimension*. In this particular example, the minimum embedding dimension is 4 $(x_1, \dot{x}_1, x_2, \dot{x}_2)$. Alternatively, one may argue that in this case we may avoid an ambiguity and separate dimensions encoding $\ddot{x}_1 = f_1(x_1, \dot{x}_1)$, though it is possible in this particular case, it will lead to the necessity to analyze 5 state variables $(x_1, \dot{x}_1, \ddot{x}_1, x_2, \dot{x}_2)$. Furthermore, to preserve a spatial correlation pattern between x_1 and x_2 the decoupled systems should be synchronized by an external mechanism.

TABLE VI
APPENDIX I. COMPARISON OF THE PROPOSED METHOD WITH [IJSPEERT ET AL., 2001]

GMR-based method proposed in this paper:

a single multidimensional system is running to control several DOFs

$$\dot{\mathbf{x}} = \hat{f}(\mathbf{x})$$
$$\hat{f}(\mathbf{x}) \triangleq \sum_{k=1}^{K} h_k(\mathbf{x})(\mu_{k,\dot{\mathbf{x}}} + \Sigma_{k,\dot{\mathbf{x}}\mathbf{x}}\Sigma_{k,\mathbf{x}}^{-1}(\mathbf{x} - \mu_{k,\mathbf{x}}))$$

where $\mathbf{x} \in \mathbb{R}^N$; $\Sigma_{k,\dot{\mathbf{x}}\mathbf{x}},\ \Sigma_{k,\mathbf{x}} \in \mathbb{R}^{N\times N}$ are estimated matrices
$\mu_{k,\dot{\mathbf{x}}},\ \mu_{k,\mathbf{x}} \in \mathbb{R}^N$ are estimated vectors

LWPR-based method proposed in [Ijspeert et al., 2001] (see also Figure 2):

the velocity along each DOF $\dot{x}$ is defined by a linear combination of two velocities $\dot{z}$ and $\dot{\nu}$, according to:

$$\dot{x} = \dot{z} + \hat{f}^*(\nu)\dot{\nu}$$

$$\hat{f}^*(\nu) \triangleq \frac{\sum_{k=1}^{K} \Psi_k(\nu)\omega_k}{\sum_{k=1}^{K} \Psi_k(\nu)}$$

where $x,\ z,\ \nu \in \mathbb{R}$
$\Psi_k(\nu) = \exp\frac{(\nu - c_k)^2}{2\sigma_k^2}$, $\omega_k \in \mathbb{R}$.

The variables z and ν are governed by two dynamical system:

(S1)	(S2)
$\ddot{\nu} = \alpha_v(\beta_v(g - \nu) - \dot{\nu})$	$\ddot{z} = \alpha_z(\beta_z(g - y) - \dot{z})$ $\dot{y} = \dot{z} + \hat{f}^*(\nu)\dot{\nu}$
where $g,\ v \in \mathbb{R}$, $\alpha_v,\ \beta_v \in \mathbb{R}$ are known constants	where $y, z \in \mathbb{R}$; $\alpha_z, \beta_z \in \mathbb{R}$ are known constants

Comparison between GMR-based $f(\mathbf{x})$ and LWPR-based $f^*(\nu)$:

Function $f^*(\nu)$ represents a uni-dimensional simplified version of a function $f(\mathbf{x})$, indeed, weights $h_k(\mathbf{x})$ have the same form of Gaussians as $\Psi_k(\nu)$, further instead of introducing a variable components $(\mu_{k,\dot{\mathbf{x}}} + \Sigma_{k,\dot{\mathbf{x}}\mathbf{x}}\Sigma_{k,\mathbf{x}}^{-1}(\mathbf{x} - \mu_{k,\mathbf{x}}))$, LWPR considers merely constants ω_k, which is equivalent to using solely $\mu_{k,\dot{\mathbf{x}}}$.

Weights ω_k are tuned so to minimize a mean-square error between the velocity $\dot{y}$ and a demonstrated velocity profile.

Note, that according to the LWPR-based method [Ijspeert et al., 2001] the function $\hat{f}^*(\nu)$ modulating the velocity $\dot{x}$ does not depend on the actual position x, but instead depends on the internal state ν and, therefore, does not introduce a feedback loop. Practically it means that the only term adapting during perturbations is $\dot{z}$, while $\hat{f}^*(\nu)\dot{\nu}$ remains the same and may deform a trajectory.

The system (S1) is a spring and damper system, which attracts a trajectory ν towards the target g following a straight line path.

The system (S2) is a perturbed spring and damper system: initially it starts to go exactly as the system (S1), but due to the perturbed velocity $\dot{y}$, it departs from the straight line trajectory ν; the source of perturbation is a component $\hat{f}^*(\nu)\dot{\nu}$.

VIII. ACKNOWLEDGMENTS

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

The authors warmly thank Prof. Auke Jan Ijspeert for advices and support toward the research work developed here.

REFERENCES

J. Aleotti, S. Caselli, and M. Reggiani. Evaluation of virtual fixtures for a robot programming by demonstration inter-

face. *Transactions on Systems, Man, and Cybernetics*, 35(4): 536–545, 2005.

R. Andersson. Aggressive trajectory generator for a robot ping-pong player. volume 9, pages 15–21, Feb 1989. doi: 10.1109/37.16766.

M. Aoki. *State Space Modeling of Time Series*. Springer-Verlag, 1990.

X. Bai, X.-S. Yang, and H. Li. Estimates of the region of attraction of continuous-time cascade systems. *Journal of Mathematical Control and Information*, 24(4)::483–491, 2007.

R. Bellman. *Dynamic Programming*. NJ:Princeton Univ. Press, 1957.

A. Billard, S. Calinon, R. Dillmann, and S. Schaal. Robot Programming by Demonstration. In *Handbook of Robotics*, volume chapter 59. MIT Press, 2008.

M. D. Buhmann. *Radial Basis Functions: Theory and Implementations*. Cambridge University Press, 2003.

S. Calinon and A. Billard. What is the Teacher's Role in Robot Programming by Demonstration? - toward Benchmarks for Improved Learning. *Interaction Studies. Special Issue on Psychological Benchmarks in Human-Robot Interaction*, 8 (3), 2007. doi: NA.

S. Calinon and A. Billard. A Probabilistic Programming by Demonstration Framework Handling Constraints in Joint Space and Task Space. In *In the Proceedings of the International Conference of Intelligent Robots and Systems*, 2008. URL http://iros2008.inria.fr/.

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, Part B. Special issue on robot learning by observation, demonstration and imitation*, 37(2):286–298, 2007. doi: NA.

T. L. Carroll. A nonlinear dynamics method for signal identification. *Chaos*, 17, 2007.

F. Chamroukhi, A. Same, G. Govaert, and P. Aknin. Time series modelling by a regression approach based on a latent process. *Neural Networks.*, 22:593–602, 2009.

M. B. Crutchfield J.P. Equation of motion from a data series. *Complex Systems*, 1:417–452, 1987.

K. D., 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*, 27(7):761–784, 2008.

M. Deisenroth, C. Rasmussen, and J. Peters. Gaussian process dynamic programming. *Neurocomputing*, 72:1508–1524, 2009.

Y. Demiris and B.Khadhouri. Hierarchical attentive multiple models for execution and recognition of actions. *Robotics and Autonomous Systems*, 54:361–369, 2006.

A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of Royal Statistic Society*, 39:1–38, 1977. doi: http://dx.doi.org/10.2307/2984875. URL http://dx.doi.org/10.2307/2984875.

R. Diankov and J. James Kuffner. Randomized statistical path planning. In *Proceedings of IEEE/RSJ International Conference on Robots and Systems (IROS)*, 2007.

K. Dixon and P. Khosla. Trajectory representation using sequenced linear dynamical systems. *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, 4:3925–3930, 26-May 1, 2004. ISSN 1050-4729.

R. Genesio, M. Tartaglia, and A. Vicino. On the estimation of asympototic stability regions: state of the art and new proposals. *IEEE Transaction on Automatic Control*, 30(8): 1985, 1985.

P. Giesl. Construction of a local and global lyapunov function using radial basis functions. *Journal of Applied Mathematics*, 73(5)::782–802, 2008.

E. Gribovskaya and A. Billard. Combining Dynamical Systems Control and Programming by Demonstration for Teaching Discrete Bimanual Coordination Tasks to a Humanoid Robot. In *IEEE/ASM International Conference on Human-Robot Interaction*, 2008.

F. Guenter, M. Hersch, S. Calinon, and A. Billard. Reinforcement Learning for Imitating Constrained Reaching Movements. volume 21, pages 1521–1544, 2007. doi: NA.

W. Hardle. *Smoothing Techniques with Implementation in Statistics*. NY:Springer, 1991.

M. Hersch, F. Guenter, S. Calinon, and A. Billard. Dynamical System Modulation for Robot Learning via Kinesthetic Demonstrations. *IEEE Transactions on Robotics*, 1, 2008. Accepted.

P. A. Hinrichsen D. *Mathematical Systems Theory*. Springer Berlin, 2000.

J. Hwang, R. Arkin, and D.-S. Kwon. Mobile robots at your fingertip: Bezier curve on-line trajectory generation for supervisory control. *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, 2:1444–1449, Oct. 2003.

A. Ijspeert and A. Crespi. Online trajectory generation in an amphibious snake robot using a lamprey-like central pattern generator model. *Robotics and Automation, 2007 IEEE International Conference on*, 1:262–268, April 2007. ISSN 1050-4729. doi: 10.1109/ROBOT.2007.363797.

A. Ijspeert, J. Nakanishi, and S. Schaal. Trajectory formation for imitation with nonlinear dynamical systems. *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, 2:752–757 vol.2, 2001.

A. Ijspeert, J. Nakanishi, and S. Schaal. Learning rhythmic movements by demonstration using nonlinear oscillators. In *In Proceedings of the IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS2002*, pages 958–963, 2002.

H. Khalil. *Nonlinear systems*. Prentice Hall Upper Saddle River, NJ, 1996.

J. Kuffner, S. Kagami, and I. M. I. H. Nishiwaki, K. Dynamically-stable motion planning for humanoid robots. *Autonomous Robots*, 12(1):105–118, 2002.

E. T. Y. Lee. Comments on some b-spline algorithms. *Computing (Springer-Verlag)*, 36:229–238, 1986.

L. Ljung. State of the art in linear system identification: Time and frequency domain methods. *Proceeding of the 2004 American Control Conference*, 1:650–661, 2004.

M. Loccufier and E. Noldus. A new trajectory reversing method for estimating stability regions of autonomous non-

linear systems. *Nonlinear Dynamics*, 21(3):265–288, 2000.

G. McLahlan and D. Peel. *Finite Mixture Models*. NY:Wiley, 2000.

A. Moore. *Efficient memory-based learning for robot control*. PhD thesis, University of Cambridge, 1990.

H.-G. Muller. *Nonparametric Regression Analysis of Longitudinal Data*. Berlin:Springer, 1988.

D. Nguyen-Tuong, M. Seeger, and J. Peters. Local gaussian process regression for real time online model learning and control. In *Proceedings of the Conference on Advances in Neural Information Processing Systems*, 2008.

J. Park and I. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2):246 – 257, 1991.

M. Priestley. State-dependent models: A general approach to non-linear time series analysis. *Journal of Time Series Analysis*, 1, 1980.

L. Righetti, J. Buchli, and A. Ijspeert. Dynamic hebbian learning in adaptive frequency oscillators. *Physica D*, 216(2):269–281, 2006. URL http://dx.doi.org/10.1016/j.physd.2006.02.009.

R. W. L. Ryoung K. Lim, Minh Q. Phan. State-space system identification with identified hankel matrix. Technical report, Department of Mechanical and Aerospace Engineering Technical Report No.3045, Princeton University, Princeton, NJ., 1998.

S. Schaal and C. Atkeson. Robot juggling: implementation of memory-based learning. *Control Systems Magazine, IEEE*, 14(1):57–71, Feb 1994. ISSN 0272-1708. doi: 10.1109/37.257895.

S. Schaal and C. G. Atkeson. Constructive incremental learning from only local information. *Neural Computation*, 10(8):2047–2084, 1998. URL citeseer.ist.psu.edu/schaal97constructive.html.

S. Schaal, S. Kotosaka, and D. Sternard. Nonlinear dynamical systems as movement primitives. In *Proceedings of the International Conference on Humanoid Robotics*, 2001.

S. Schaal, A. Ijspeert, and A. Billard. Computational Approaches to Motor Learning by Imitation. *Philosophical transactions: biological sciences*, 358(1431):537–547, 2003. doi: NA.

G. Schoner and C. Santos. Control of movement time and sequential action through attractor dynamics: A simulation study demonstration object perception and coordination. In *Symposium on Intelligent Robotic Systems*, 2001.

D. Sternad and D. Schaal. Segmentation of endpoint trajectories does not imply segmented control. *Experimental Brain Research*, 124:118–136, 1999.

H. G. Sung. *Gaussian Mixture Regression*. PhD thesis, Rice Univercity, Huston, Texas, 2004.

H. Tomohisa, W. M. Haddad, and H. Naira. Neural network adaptive control for a class of nonlinear uncertain dynamical systems with asymptotic stability guarantees. *IEEE Transactions on Neural Networks*, 19:80–90, 2008.

D. Travis, B. T. Thumati, and S. Jagannathan. Optimal control of unknown affine nonlinear discrete-time systems using offline-trained neural networks with proof of convergence. *Neural Networks*, 22:851–860, 2009.

A. Ude, C. Atkeson, and M. Riley. Programming full-body movements for humanoid robots by observation. *Robotics and Autonomous Systems*, 47:93–108, 2004.

S. Wang, H. Luo, C. Yue, and X. Liao. Parameter identification of chaos system based on unknown parameter observer. *Physics letters. A*, 372:2603–2607, 2008.

H. Wei and S. Amari. Dynamics of learning near singularities in radial basis function networks. *Neural Networks*, 21: 981–1005, 2008.

N. Xie and H. Leung. Blind identification of autoregressive system using chaos. *IEEE Transactions on Circuits and Systems*, 52:1953–1965, 2005.

K. Yamane, J. J. Kuffner, and J. K. Hodgins. Synthesizing animations of human manipulation tasks. *ACM Trans. Graph.*, 23(3):532–539, 2004. ISSN 0730-0301. doi: http://doi.acm.org/10.1145/1015706.1015756.

K. Yokoi, E. Yoshida, and H. Sanada. Unified motion planning of passing under obstacles with humanoid robots. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 1185–1190, May 2009. doi: 10.1109/ROBOT.2009.5152797.

E. Yoshida, M. Poirier, J. Laumond, O. Kanoun, F. Lamiraux, R. Alami, and K. Yokoi. Whole-body motion planning for pivoting based manipulation by humanoids. In *Proceedings of IEEE International Conference on Robotics and Automation*, 2008.

R. Zollner, T. Afour, and D. R. Programming by demonstration: Dual-arm manipulation tasks for humanoid robots. In *Proceedings of the International Conference on Intelligent Robots and Systems*, 2004.