Statistical learning by imitation of competing constraints in joint space and task space

Sylvain Calinon Aude Billard

Learning Algorithms and Systems Laboratory (LASA), Ecole Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland

{*sylvain.calinon, aude.billard*}*@epfl.ch*

Abstract

We present a probabilistic architecture for solving generically the problem of extracting the task constraints through a Programming by Demonstration (PbD) framework and for generalizing the acquired knowledge to various situations. In previous work, we proposed an approach based on Gaussian Mixture Regression (GMR) to find a controller for the robot reproducing the statistical characteristics of a movement in joint space and in task space through Lagrange optimization. In this paper, we develop an alternative procedure to handle simultaneously constraints in joint space and in task space by combining directly the probabilistic representation of the task constraints with a solution to Jacobian-based inverse kinematics. The method is validated in manipulation tasks with two 5 DOFs Katana robotic arms displacing a set of objects.

Keywords: Robot programming by demonstration, learning by imitation, kinesthetic teaching, Gaussian mixture regression, inverse kinematics

1 INTRODUCTION

Robot Programming by Demonstration (PbD) covers methods by which a robot learns new skills through human guidance. In previous work, we presented an approach to teach gestures to a HOAP-3 humanoid robot by providing a set of demonstrations performed in various situations. Through the use of *Gaussian Mixture Model* (GMM), the robot could extract autonomously the statistical characteristics of the set of trajectories captured through the demonstrations [1, 2]. Then, *Gaussian Mixture Regression* (GMR) was used to retrieve a generalized version of the trajectories either in joint space (characterized by a set of postures changing through time), or in task space (characterized by the 3D Cartesian position of the hand relative to the objects in the scene). To find a controller for the robot that takes into account constraints both in joint space and in task space (as well as the kinematic redundancy of the humanoid arm), we previously proposed two approaches: (1) a method based on Lagrange optimization [1]; and (2) a geometric inverse kinematics approach for a 4 DOFs humanoid arm by representing the motion of the arm as the 3D Cartesian path of the hand with an additional parameter representing the elevation of the elbow with respect to a vertical plane [2]. Even if these approaches provided solutions for the reproduction of a set of constraints in different data spaces, they still lacked generality when the skill required to handle simultaneously task space and joint space variables. Indeed, in [1], a metric of imitation performance had to be analytically derived to find an optimal controller for the reproduction. In [2], the geometric approach could not be directly applied to more complex robot architectures such as the 5 DOFs *Katana* robots that we consider here. Here, we propose a robot programming by demonstration approach that combines the statistical representation of the motion (encoded in GMM) with the local properties of a Jacobian-based solution to inverse kinematics. The approach allows us to simultaneously handle constraints on multiple objects in task space and in joint space, and can be used generically for different robot architectures.

1.1 Related work

Generic approaches to transfer new skills to a robot are those that allow the robot to extract automatically what are the important features characterizing the skill and to search for a controller that optimizes the reproduction of these characteristic features [3]. A key concept at the bottom of these approaches is that of determining a *metric of imitation performance*. One must first determine the metric, i.e. determine the weights one must attach to reproducing each of the components of the skill. It is then possible to find an optimal controller for imitation by trying to minimize this metric (e.g., by evaluating several reproduction attempts or by deriving the metric to find an optimum). The metric acts as a cost function for the reproduction of the skill [4]. In other terms, a metric of imitation provides a way of expressing quantitatively the user's intentions during the demonstrations and to evaluate the robot's faithfulness at reproducing those. To learn the metric (i.e. infer the task constraints), one common approach consists of creating a model of the skill based on several demonstrations performed in slightly different conditions. This generalization process consists of exploiting the variability inherent to the various demonstrations to extract which are the essential components of the task. These essential components should be those that remain invariant across the various demonstrations.

A large body of work explored the use of a symbolic representation to both the learning and the encoding of skills and tasks, see e.g. [5, 6]. The main advantage of a symbolic approach is that high-level skills (consisting of sequences or hierarchies of symbolic cues) can be learned efficiently through an interactive process. However, because of the symbolic nature of their encoding, these methods rely on a large amount of prior knowledge to predefine the important cues and to segment those efficiently.

Another body of work focusses on representing the task constraints at a trajectory level to avoid putting too much prior knowledge in the controllers required to reproduce a skill. Following this approach, Ude *et al* [7] use spline smoothing techniques to deal with the uncertainty contained in several demonstrations of motion performed in *joint space* or in *task space*. The *Mimesis Model* [8] follows an approach in which a *Hidden Markov Model* (HMM) is used to encode a set of trajectories, and where multiple HMMs can be used to retrieve new generalized motions based on a stochastic process. In [9], the variability across demonstrations performed by several demonstrators is used to quantify the accuracy required to achieve a *Pick & Place* task. The different trajectories form a boundary region that is then used to define a range of acceptable trajectories.

1.2 Proposed approach

Several regression techniques based on a probabilistic representation of the dataset such as *Locally Weighted Regression* (LWR) [10, 11] or *Gaussian Process Regression* (GPR) [12] were proposed in robotics to generalize over a set of demonstrations. Our approach follows a similar strategy by using *Gaussian Mixture Model* (GMM) and *Gaussian Mixture Regression* (GMR) [13] to respectively encode a set of trajectories and retrieve a smooth generalized version of these trajectories with associated variances, where the dataset is encoded in a compact form learned through *Expectation-Maximization* (EM) algorithm.[1]

To control redundant manipulators in task space, several inverse kinematics solutions based on local resolution methods capable of handling multiple constraints simultaneously were proposed [14]. Grochow *et al* [15] proposed an alternative strategy for computer graphics animation of avatars by resolving the redundancy of the inverse kinematics problem through the observation of a set of human motions which guided the search of a solution that looks similar to natural human gestures. Our approach follows in essence a similar strategy by combining several constraints expressed both in task space and in joint space. In the proposed approach, the search for an inverse kinematics solution is facilitated by the user who implicitly provides in his/her demonstrations possible solutions for the resolution of the task, thus restricting the search space of the robot for inverse kinematics solutions. To do so, the robot first computes several inverse kinematics solutions solving the different constraints in task space, and then combines these constraints with the ones represented initially in joint space.

2 GAUSSIAN MIXTURE REGRESSION

We describe in this section the *Gaussian Mixture Regression* (GMR) process used to reproduce a learned skill, which is based on the theorem of Gaussian conditioning and on the linear combination properties of Gaussian distributions.

We define a trajectory as a set of position data (in joint space or task space) indexed with time. Time t is considered here as an input variable $\xi^{\mathcal{I}} = t$ used to retrieve an expected position at that time. The output variable is then either $\xi^{\mathcal{O}} = \theta$ in joint space or $\xi^{\mathcal{O}} = x$ in task space. The joint probability distribution $\mathcal{P}(\xi^{\mathcal{I}}, \xi^{\mathcal{O}})$ is first modeled by a *Gaussian Mixture Model* (GMM). The retrieval process then consists of estimating $\mathbb{E}\left[\mathcal{P}(\xi^{\mathcal{O}}|\xi^{\mathcal{I}})\right]$, with associated constraints estimated by $\operatorname{cov}\left(\mathcal{P}(\xi^{\mathcal{O}}|\xi^{\mathcal{I}})\right)$. We use here the notations $\mathbb{E}[\cdot]$ and $\operatorname{cov}(\cdot)$ to express respectively expectation and covariance. The

[1]The advantages of GMR for the proposed application will be discussed in Section 6.

Table 1: Gaussian Mixture Regression process.

A dataset of N datapoints of D dimensions is encoded in a *Gaussian Mixture Model* (GMM) of K Gaussians. The probability that a datapoint $\xi = [\xi^{\mathcal{I}}, \xi^{\mathcal{O}}]$ belongs to the GMM is defined by

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

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. Input and outputs components are represented separately as

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

For a given input variable $\xi^{\mathcal{I}}$ and a given Gaussian distribution k, the expected distribution of $\xi^{\mathcal{O}}$ is defined by

$$\mathcal{P}(\xi^{\mathcal{O}} | \xi^{\mathcal{I}}, k) \sim \mathcal{N}(\hat{\xi}_k, \hat{\Sigma}_k) \ , \quad \text{where} \quad \begin{aligned} \hat{\xi}_k &= \mu_k^{\mathcal{O}} + \Sigma_k^{\mathcal{OI}} (\Sigma_k^{\mathcal{I}})^{-1} (\xi^{\mathcal{I}} - \mu_k^{\mathcal{I}}), \\ \hat{\Sigma}_k &= \Sigma_k^{\mathcal{O}} - \Sigma_k^{\mathcal{OI}} (\Sigma_k^{\mathcal{I}})^{-1} \Sigma_k^{\mathcal{IO}}. \end{aligned}$$

By considering the complete GMM, the expected distribution of $\xi^{\mathcal{O}}$, when $\xi^{\mathcal{I}}$ is known, can be estimated as

$$\mathcal{P}(\xi^{\mathcal{O}} | \xi^{\mathcal{I}}) \sim \sum_{k=1}^{K} h_k \, \mathcal{N}(\hat{\xi}_k, \hat{\Sigma}_k),$$

where $h_k = \mathcal{P}(k | \xi^{\mathcal{I}})$ is the probability that the Gaussian distribution k is responsible for $\xi^{\mathcal{I}}$

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

By using the linear transformation property of Gaussian distributions [16], the conditional expectation of $\xi^{\mathcal{O}}$, given $\xi^{\mathcal{I}}$, can be approximated by a single Gaussian distribution $\mathcal{N}(\hat{\xi}, \hat{\Sigma})$ with parameters

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

generic *Gaussian Mixture Regression* (GMR) process is presented in Table 1. Equation (1) defines the conditional expectation of a multivariate normal distribution by estimating it as a least squares estimate, i.e., the probability function $\mathcal{P}(\xi^{\mathcal{O}}|\xi^{\mathcal{I}})$ is represented by a single Gaussian distribution.

Gaussian Mixture Regression (GMR) can be used to retrieve smooth generalized trajectories with associated covariance matrices describing the variations and correlations across the different variables. In a generic regression problem, one is given a set of *predictor* variables $\xi^{\mathcal{I}} \in \mathbb{R}^p$ and *response* variables $\xi^{\mathcal{O}} \in \mathbb{R}^q$. The aim of regression is to estimate the conditional expectation of $\xi^{\mathcal{O}}$ given $\xi^{\mathcal{I}}$, on the basis of a set of observations $\{\xi^{\mathcal{I}}, \xi^{\mathcal{O}}\}$. For the particular example of trajectory learning, on the basis of a set of observations $\{\xi^{\mathcal{I}}, \xi^{\mathcal{O}}\}$, where $\xi^{\mathcal{O}}$ represents position vectors at time steps $\xi^{\mathcal{I}} = t$, the aim of the regression process is to estimate the conditional expectation of $\xi^{\mathcal{O}}$ at each time step to retrieve a generalized trajectory. Thus, GMR offers a way of extracting a single generalized trajectory made up from a set of trajectories used to train the model, where the generalized trajectory is not part of the dataset but instead encapsulates all of its essential features. As the different demonstrations are performed at different speed or have temporal variations, *Dynamic Time Warping* (DTW) is used as a pre-processing step to re-align temporally the trajectories, see [1].

Compared to traditional regression approaches, the regression function is not approximated directly. Instead, the joint density of the set of trajectories is first estimated by a model from which the regression function is derived. The regression process generates both a mean response estimate $\hat{\xi}$ and a covariance response estimate $\hat{\Sigma}$, conditioned on the predictor variables $\xi^{\mathcal{I}}$. To learn trajectories, $\hat{\xi}$ is used as a generalized trajectory, while $\hat{\Sigma}$ is used to extract the constraints on this trajectory. It thus permits to evaluate the covariance information not only at specific positions but continuously along the movement.

Thus, the modeling phase becomes the important part of the algorithm in terms of processing, while the regression phase is processed very quickly, which is advantageous because the reproduction of smooth trajectories is fast enough to be used at any appropriate time by the robot, while the joint density modeling is computed in a phase of interaction that does not particularly need fast computation. Indeed, the estimation of $\mathbb{E}\left[\mathcal{P}(\xi^{\mathcal{O}}|\xi^{\mathcal{I}})\right]$ is simply computed by a weighted sum of linear models. For regression, the main advantage of a model-based approach over a data-driven approach is that the computation time required for reproduction does not increase with the number of demonstrations provided to the robot, which is a particularly important property for lifelong learning robots.

Although the theoretical considerations of GMR have been studied in the machine learning literature several years ago [13], the theory has come out with only few applications, which is surprising since GMM and associated *Expectation-Maximization* (EM) learning algorithms are very well established in various practical fields of research. The approach is generic in the sense that it can be used both for recognition and reproduction, and allows to change on-the-fly the input and output variables to consider during reproduction (e.g., to deal with missing variables). It is easy to implement, and satisfies robustness and smoothness criterions that are common to different fields of research including robotics.[2]

[2]Matlab and C++ sourcecodes for the encoding and reproduction processes are available from `http://programming-by-demonstration.org`.

The number of Gaussians in the GMM determines the compromise for GMR between having an accurate estimation of the response and having a smooth response, known as the *bias-variance tradeoff*. An appropriate model complexity must thus be determined in order to: (1) have a low bias between the estimate and the real values (accuracy of the estimation with respect to the observed data); and (2) have a low variance on the estimate (smoothness of the estimation). It is estimated here through *Bayesian Information Criterion* (BIC), see [1].

3 HANDLING MULTIPLE CONSTRAINTS DURING REPRODUCTION

We described in previous section a probabilistic method to extract task constraints from a set of trajectories and to reproduce a generalized version of these trajectories. We now consider the problem of finding an optimal controller for the robot when multiple independent constraints are considered and represented separately in different GMMs (e.g., by considering actions on two different objects). In this situation, the joint probability (i.e., the probability of two events in conjunction) can be computed by estimating $\mathbb{E}\left[\mathcal{P}(\hat{\xi}^{(1)}) \cdot \mathcal{P}(\hat{\xi}^{(2)})\right]$ (independence assumption). By using the standard statistical properties of Gaussian distributions, namely linear transformation, product and conditional distribution, this estimation can be handled easily.

We show here that this estimation can also be interpreted as the minimization of a cost function measuring the similarity between the demonstrated and reproduced trajectories (optimization of a metric of imitation). We take the perspective that a task can be represented as a set of "loose" constraints, namely, constraints represented as a set of optimal values (the centers of the Gaussians) that need to be satisfied as best as possible within a range given by the covariance matrices. We define a metric of imitation as a fixed and generic cost function whose parameters are learned through observation (i.e. the form of the cost function is discovered by imitation).[3] A controller for the robot is then determined by optimizing this cost function under strict constraints defined by inverse kinematics.

We consider the generic case where one looks for an optimal controller that combines constraints in joint space and task space. To treat simultaneously end-effector paths x and joint angles θ in a probabilistic framework, we use the fact that $\dot{\theta}$ and $\dot{x}$ are kinematically constrained. Velocities are first estimated through Euler numerical integration[4]

$$\dot{x}_t = \frac{1}{\Delta t}(x_t - x_{t-\Delta t}), \qquad \dot{\theta}_t = \frac{1}{\Delta t}(\theta_t - \theta_{t-\Delta t}),$$

where t refers to the time elapsed from the beginning of the demonstration. The generalized joint angle velocities $\hat{\dot{\theta}}$ and generalized end-effector velocities $\hat{\dot{x}}$ (both retrieved through the GMM/GMR approach presented in Section 2) can be mutually exclusive in the imitator workspace, due to the varying situations

[3]As a metric involves the notion of distance and a cost function implicitly refers to optimization, we will use both terms interchangeably in the remaining of the paper.

[4]Note however that other numerical methods for ordinary differential equations can similarly be used here [17].

between demonstrations and the reproduction attempt, or simply because the generalization processes are performed separately. To simultaneously fulfill constraints in joint space and task space, the product properties of normal distributions can be used together with local inverse kinematics properties to find a controller satisfying the two constraints.

3.1 Method based on the optimization of a metric of imitation

Let $\hat{\dot{\theta}}$ and $\hat{\dot{x}}$ be respectively the generalized joint angle velocities and the generalized end-effector velocities in Cartesian space. Let $\dot{\theta}$ and $\dot{x}$ be the candidate velocities for reproducing the motion. An optimal controller can be determined by solving the constrained optimization problem

$$\min_{\dot{\theta},\dot{x}} \quad (\dot{\theta}-\hat{\dot{\theta}})^\top\, W^\theta\, (\dot{\theta}-\hat{\dot{\theta}}) \;+\; (\dot{x}-\hat{\dot{x}})^\top\, W^x\, (\dot{x}-\hat{\dot{x}}) \qquad \text{u.c.} \quad \dot{x} = J\dot{\theta}, \tag{2}$$

where J is the Jacobian matrix at posture θ_t. $W^\theta \in \mathbb{R}^{n\times n}$ and $W^x \in \mathbb{R}^{m\times m}$ are semi-definite positive diagonal matrices serving as coefficient indicating the respective influence that one should give to the desired joint angles and end-effector location. Coherence is then enforced by finding the velocities $(\dot{x},\dot{\theta})$ that will bring the system closest to $(\hat{\dot{x}},\hat{\dot{\theta}})$.

A solution can be obtained using Lagrange multipliers. The quantity to minimize can be expressed as

$$L(\dot{\theta},\dot{x}) \quad = \quad (\dot{\theta}-\hat{\dot{\theta}})^\top\, W^\theta\, (\dot{\theta}-\hat{\dot{\theta}}) \quad + \quad (\dot{x}-\hat{\dot{x}})^\top\, W^x\, (\dot{x}-\hat{\dot{x}}) \quad - \quad \lambda^\top(\dot{x}-J\dot{\theta}),$$

where λ is the vector of Lagrange multipliers. After differentiating with respect to $\dot{\theta}$ and $\dot{x}$, and setting the equation to zero, one gets

$$\dot{\theta} = (W^\theta + J^\top W^x J)^{-1}(J^\top W^x \hat{\dot{x}} + W^\theta \hat{\dot{\theta}}). \tag{3}$$

An alternate representation of (3) developed in [18] can be formulated as

$$\dot{\theta} = \hat{\dot{\theta}} + (W^\theta)^{-1}J^\top\left((W^x)^{-1} + J(W^\theta)^{-1}J^\top\right)^{-1}(\hat{\dot{x}} - J\hat{\dot{\theta}}). \tag{4}$$

Although simpler than (4), (3) may be disadvantageous from an implementation perspective, see [18]. Indeed, using $(W^\theta)^{-1}$ and $(W^x)^{-1}$ instead of W^θ and W^x renders it possible to avoid infinity measures when dealing with purely angular controllers. It is indeed easier to handle $(W^\theta)^{-1}$ (or $(W^x)^{-1}$) as equal to zero during computation. Moreover, this formulation is faster as it requires a matrix inversion of degree m, whereas (3) requires the inversion of a matrix of degree $n > m$ (for redundant manipulators). The parameters W^θ and W^x control the influence of each of the sub-controllers. By setting W^x to zero, one obtains a pure joint angle controller and by setting W^θ to zero, the result becomes a pure end-effector location controller. The results given by (4) or (3) are in fact generalizations of the *Weighted Least Norm* solution and of the *Weighted Damped Least Square* solution of the inverse kinematics problem, see [18]. Indeed, by setting $\hat{\dot{\theta}} = 0$ (least-norm controller) and $(W^x)^{-1} = 0$ in (4), one obtains the *Weighted Least Norm* method which makes use of the Moore-Penrose pseudoinverse of the Jacobian to compute the adequate joint angle velocities [19]. Furthermore, by setting $\hat{\dot{\theta}} = 0$ in (3), one obtains the *Weighted Damped Least Square* solution to avoid singularities, see [18].

3.2 Direct computation method

Instead of analytically deriving a metric of imitation, the same result can be retrieved through the product property of normal distributions. Let us assume that the constraints in task space are defined by $x \sim \mathcal{N}(\hat{x}, \hat{\Sigma}^{\dot{x}})$, and that the constraints in joint space are defined by $\dot{\theta} \sim \mathcal{N}(\hat{\dot{\theta}}, \hat{\Sigma}^{\dot{\theta}})$. When using the linear transformation property of normal distributions and the inverse kinematics relation $\dot{\theta} = J^{\dagger}\dot{x}$ where $J^{\dagger}$ denotes the pseudoinverse of the Jacobian, the constraints in task space can be represented in joint space as

$$\dot{\theta}' \sim \mathcal{N}\left(J^{\dagger}\hat{\dot{x}}, J^{\dagger}\hat{\Sigma}^{\dot{x}}(J^{\dagger})^{\top}\right). \tag{5}$$

Based on the product property of normal distributions [16], the product of the two Gaussian distributions describing $\dot{\theta}$ and $\dot{\theta}'$ is equivalent (up to a scaling factor) to a Gaussian distribution $\mathcal{N}(\dot{\theta}, \Sigma^{\dot{\theta}})$ with parameters[5]

$$\begin{aligned} \dot{\theta} &= \left((\hat{\Sigma}^{\dot{\theta}})^{-1} + J^{\top}(\hat{\Sigma}^{\dot{x}})^{-1}J\right)^{-1} \left((\hat{\Sigma}^{\dot{\theta}})^{-1}\hat{\dot{\theta}} + J^{\top}(\hat{\Sigma}^{\dot{x}})^{-1}\hat{\dot{x}}\right), \\ \Sigma^{\dot{\theta}} &= \left((\hat{\Sigma}^{\dot{\theta}})^{-1} + J^{\top}(\hat{\Sigma}^{\dot{x}})^{-1}J\right)^{-1}. \end{aligned} \tag{6}$$

By considering $W^{\theta} = (\hat{\Sigma}^{\theta})^{-1}$ and $W^{x} = (\hat{\Sigma}^{x})^{-1}$, we then see that (3) and (6) provide the same result. Compared to the derivation of a cost function through Lagrange multipliers, the direct computation method has the advantage of not only determining an optimal controller satisfying several constraints but also to compute the resulting constraints for this controller, represented as $\Sigma^{\dot{\theta}}$ in (6). It should be noted that, instead of considering $\dot{x} = J\dot{\theta}$ as the coherence constraint, it is possible to define a coherence constraint that takes advantages of the robot kinematic redundancy. One can, for instance, consider an optimization term that does not modify the coherence constraint through projection in the null space of the Jacobian.

4 EXPERIMENTAL SETUP

The setup of the experiment is presented in Figure 1. Two 5-DOFs *Katana* robots from *Neuronics*, characterized by a repeatability of ± 0.1 mm and a maximum speed of $68°$/sec, are used for the experiment. A sixth motor controls the opening and closing status of the gripper, which is generated through a binary signal generalized over multiple demonstrations through *Bernoulli Mixture Regression* as described in [20]. Each motor is equipped with encoders permitting the user to move the robot manually while registering joint angle information (see Figure 1). During this process, the position of the end-effector is also computed through direct kinematics. A stereoscopic vision system based on two webcams of 320×240 pixels is used to track a set of objects in 3D Cartesian space based on tracking in *YCbCr* color space of colored patches attached to the objects (only *Cb* and *Cr* are used to be robust to changes in luminosity), where each object to track is pre-defined in a calibration phase.

[5] Note that, for redundant robot kinematics, $JJ^{\dagger} = I$ and $J^{\dagger}\hat{\Sigma}^{\dot{x}}(J^{\dagger})^{\top}$ is not full rank. In this situation, a pseudoinverse needs to be used instead of an inverse, yielding $\left(J^{\dagger}\hat{\Sigma}^{\dot{x}}(J^{\dagger})^{\top}\right)^{\dagger} = ((J^{\dagger})^{\top})^{\dagger}(\hat{\Sigma}^{\dot{x}})^{\dagger}(J^{\dagger})^{\dagger} = J^{\top}(\hat{\Sigma}^{\dot{x}})^{-1}J$.

Table 2: Reproduction of a skill when N objects with initial positions $\{o^{(n)}\}_{n=1}^{N}$ are detected in the robot's workspace.

```
Offline processing and initialization
• Initialization with starting posture $\theta_0 = \hat{\theta}_0$ and end-effector location $x_0 = K(\hat{\theta}_0)$, where $K$ is the direct kinematics function.
Loop for $t = \Delta t \to T$
  Loop for $n = 1 \to N$
    • Compute the expected velocities (approximated through Euler numerical differentiation) and associated covariance matrices for the constraints relative to object $n$
      $$\begin{aligned} \dot{\theta}_t^{(n)} &= J^{\dagger}(\theta_{t-\Delta t})\dot{x}_t^{(n)}, \quad \text{where} \quad \dot{x}_t^{(n)} = \frac{1}{\Delta t}\left[(o^{(n)} + \hat{x}_t^{(n)}) - x_{t-\Delta t}\right], \\ \Sigma_t^{(n)} &= J^{\dagger}(\theta_{t-\Delta t})\,\hat{\Sigma}_t^{x(n)}\,\left(J^{\dagger}(\theta_{t-\Delta t})\right)^{\top}. \end{aligned}$$
  End loop $n$
  • Compute the expected velocity and associated covariance matrix in joint space
    $$\dot{\theta}_t^{(N+1)} = \frac{1}{\Delta t}\left[\hat{\theta}_t - \theta_{t-\Delta t}\right], \qquad \Sigma_t^{(N+1)} = \hat{\Sigma}_t^{\theta}.$$
  • Compute the new posture (and associated covariance matrix) by evaluating the product $\prod_{n=1}^{N+1}\mathcal{N}(\dot{\theta}_t^{(n)}, \Sigma_t^{(n)})$, which represents the joint probability of the considered constraints
    $$\begin{aligned} \theta_t &= \theta_{t-\Delta t} + \Delta t\left[\left(\sum_{n=1}^{N+1}(\Sigma_t^{(n)})^{-1}\right)^{-1}\left(\sum_{n=1}^{N+1}(\Sigma_t^{(n)})^{-1}\dot{\theta}_t^{(n)}\right)\right], \\ \Sigma_t &= \left(\sum_{n=1}^{N+1}(\Sigma_t^{(n)})^{-1}\right)^{-1}. \end{aligned} \tag{7}$$
  • The new position of the end-effector is thus defined by $x_t = K(\theta_t)$.
End loop $t$
```

Two skills are considered in the experiment, namely setting the table by grasping a glass on a shelf and placing it on a coaster, and clearing the table by grasping the glass from the table and emptying the glass in a basin (see Figure 1). For the first task, two objects are tracked by the robot (the glass and the coaster), where the positions of the two objects can vary. For the second task, only one object is tracked by the robot, i.e., the glass is assumed to cover the coaster and the basin is assumed to be at a fixed position in the robot's workspace. The collected dataset is composed of the joint angle trajectories of the robot θ and the position of the end-effector x in the Cartesian space with respect to the initial positions of the objects in the scene. For the first task, five demonstrations of 11-dimensional trajectories are collected (5 variables describing the joint angles and 2×3 variables portraying the relative position of the end-effector with respect to the two objects), where each trajectory consists of 1000 points. For the second task, only 8-dimensional trajectories are considered due to only a single object being used.

By employing the GMM/GMR method presented in Section 2, the constraints in task space are computed with respect to the objects detected by the robot in its environment. The constraints associated with the position of the end-effector with respect to the initial position of object n are thus represented by the trajectories $\hat{x}^{(n)}$ and associated covariance matrices $\hat{\Sigma}^{x(n)}$. Similarly, the constraints in joint space correspond to $\hat{\theta}$ and $\hat{\Sigma}^{\theta}$. As discussed previously, the generalization in joint space does not necessarily coincide with the generalization in task space. To find a controller for the robot satisfying several constraints simultaneously, the direct computation method presented in Section 3.2 is used to compute an appropriate trade-off during the inverse kinematics process (product of Gaussian distributions).

The reproduction procedure is described in Table 2 and illustrated in Figure 2. By projecting the Gaussian distribution from task space to joint space through the Jacobian, it is here implicitly assumed that the nonlinear projection function can be approximated by the locally linear transformation $J^{\dagger}$, i.e., the local transformation is assumed to remain valid for the span of data represented by the covariance matrix of the Gaussian distribution [21]. A trade-off is thus computed in (7) based on the variability observed during the demonstrations to determine the respective relevance of the constraints in joint space and in task space. If one wishes to use a controller satisfying the constraints only in joint space, (7) can be replaced by $\theta_t = \theta_{t-\Delta t} + \Delta t \; \dot{\theta}_t^{(N+1)}$. Similarly, if one desires to employ a controller satisfying the constraints in task space for a specific object n, (7) can be replaced by $\theta_t = \theta_{t-\Delta t} + \Delta t \; \dot{\theta}_t^{(n)}$. For reproduction, we assume that the robot can track efficiently the desired trajectory in joint space (we do not consider the dynamics of the motion here). Note that the tracking problem is facilitated thanks to the kinesthetic teaching process (i.e., demonstrations of feasible motions provided to the robot).

5 EXPERIMENTAL RESULTS

The first and third graphs of Figure 3 show the five demonstrations for the two tasks. Figures 4 and 5 present the extracted constraints for the two tasks. The second and fourth graphs of Figure 3 provide a reproduction attempt for a new situation (with other initial positions of the objects), during which the essential features of the skill are reproduced. Figure 6 demonstrates how the constraints in joint space and task space influence the reproduction of the skill, where the final controller smoothly reproduces the essential features by adapting the extracted constraints to the new situation. At the beginning of the first task (*left*), the actions directed toward the glass are of utmost importance. Subsequently, the ones directed toward the coaster predominate. It can be seen that the controller determined by the system smoothly switches from the generalized movement directed toward the glass (see e.g., x_1 at time steps 200-500) to that directed toward the coaster (see e.g., x_1 at time steps 700-1000). For the second task (*right*), the trajectories relative to the glass are at first highly important (reaching for the glass in Cartesian space), and then give way to a controller satisfying constraints in joint space (emptying the glass by tilting it). We can observe that the controller smoothly switches from one for which constraints in task space are important (see e.g., θ_5 at time steps 200-400) to a controller where constraints in joint space are of importance (see e.g., θ_5 at time steps 600-1000).

The location of the end-effector with respect to the initial positions of the objects as well as the kinematics of the robot are hard-coded instead of being autonomously extracted by the system. This creates additional redundant information in the dataset (the joint angles can be used to reconstruct the position of the end-effector). Since manipulation tasks are considered, this information is included explicitly in the system by reason of its importance (i.e., position of the end-effector in task space) being meaningful for the skill considered and remaining general for a broad range of tasks. This way, the number of examples required to extract the important aspects of the task can be reduced, thereby alleviating the need for learning the direct kinematics of the robot.[6]

6 DISCUSSION

Section 2 presented the regression method that we used to retrieve smooth generalized versions of the demonstrated trajectories based on mixtures of Gaussians. We discuss here the similarities shared by *Gaussian Mixture Regression* (GMR) with other regression frameworks proposed in robotics.[7]

Locally Weighted Regression (LWR) was first proposed as an approach combining the simplicity of linear least squares regression with the flexibility of nonlinear regression [23], which was then applied to robot control [24, 25]. Work following this LWR approach mainly concentrated on moving on from a memory-based approach to a model-based approach, and moving on from a batch learning process to an incremental learning strategy. Schaal *et al* introduced *Receptive Field Weighted Regression* (RFWR) as a non-parametric approach to learn incrementally the fitting function without the need of storing the whole training data (i.e., without using historical data) [10]. The method is based on a receptive field approach where each receptive field has the form of a Gaussian kernel which is updated independently. The process allows to add and prune receptive fields and to deal with irrelevant inputs. In receptive fields, the parameters of the linear model are learned either incrementally or in batch mode. We have shown

[6]Note that the initial choice of the adequate task-dependent variables may not be trivial for more complex paradigms.

[7]For a review and comparisons of our approach with the different methods proposed above, the interested reader can also refer to [22, 20].

in [26] that when considering GMM, an incremental version of the *Expectation-Maximization* (EM) algorithm can be used to update the GMM parameters when new data are available. The iterative estimation process can also be regulated by bounding the covariance matrices, which is very similar to the use of a penalty factor to regulate the smoothness in the RFWR gradient descent method.

To resolve the *curse of dimensionality* [27] in RFWR, Vijayakumar *et al* suggested to improve the approach so that it can operate efficiently in high dimensional space through *Locally Weighted Projection Regression* (LWPR) [11]. Indeed, for high-dimensional data, considering the distance between a set of points in a receptive field approach is not optimal because the distance among points does not separate them well in high dimensions. By detecting locally redundant or irrelevant input dimensions, the approach suggests to locally reduce the dimensionality of the input data. The curse of dimensionality can for example be avoided by finding local projections of the input data through *Partial Least Squares* (PLS) regression [28].

The framework proposed here follows a similar strategy by considering that a set of trajectories can be represented locally by Gaussian distributions. In the experiments presented here, the trajectories were encoded in a Gaussian Mixture Model with up to 5 dimensions, which can be very efficiently handled by the *Expectation-Maximization* (EM) learning process. However, when using more complicated robots or a higher number of variables to describe the skill, it might be important to consider dimensionality reduction as a preprocessing step that can be combined with the proposed probabilistic encoding and reproduction procedures. We demonstrated in [1, 29] that the dataset can first be projected in a latent space of motion through *Principal Component Analysis* (PCA) or *Independent Component Analysis* (ICA), which acts as a global projection method to decorrelate and reduce, if possible, the dimensionality of the dataset. Then, by using GMM, the non-linearities of the trajectories are modeled by piecewise local information in the form of Gaussian distributions representing locally the variability and the correlation of the data. GMM can be viewed as an in-between local and global model, in the sense that it fits locally the input and output data using a global learning algorithm. GMR is used as a generic retrieval process to estimate multivariate output signals, given multivariate input signals.

Gaussian Process Regression (GPR) [30] shares in essence similarities with the GMM/GMR approach. It has been successfully applied to robotics to generalize over a set of demonstrations, see e.g. [12]. Similarly to *Support Vector Regression* (SVR) [31], GPR however suffers from high computational complexity which prevents its usage for large numbers of samples or online learning to date. Local variant of Gaussian processes has recently been proposed in robotics to deal with this issue [32]. Most regression models including GPR, SVR and LWPR are discriminative in that they learn $\mathcal{P}(Y|X)$ without modeling the joint distribution $\mathcal{P}(X,Y)$. We suggest in our work an alternative generative approach by first solving the most general problem of estimating $\mathcal{P}(X,Y)$. By using Gaussian Mixture Regression, the assumption is that $\mathcal{P}(X,Y)$ can be modeled by a finite set of Gaussian distributions. Regression on this model results in a mixture of linear models and naturally extends to regression with multiple outputs. Retrieving multiple outputs and their associated correlations is on the contrary not straightforward for GPR or SVR, which requires to develop other strategies to deal with multiple outputs [33].

Even if these approaches share similarities, GMR has two advantages for the applications considered here: (1) it allows to deal very flexibly with recognition and reproduction issues in a common probabilistic framework; and (2) the learning process is fast and distinct from the retrieval process. Thus, a simple learning process can be used to model the demonstrated skill during the phases of the interaction that do not require real-time computation (i.e. after the demonstrations), and a faster regression process can be used afterwards for controlling the robot in an online manner during the reproduction phase. Moreover, the multivariate input and output variables can be specified on-the-fly during reproduction, as the joint distribution has been learned previously by the model, which makes the approach robust to various types of missing variables. Further work will investigate the experimental comparisons of existing locally-weighted learning approaches for a set of representative scenarios relevant to robotics.

7 CONCLUSION AND FURTHER WORK

We presented a probabilistic framework to extract automatically the essential features characterizing a skill, by handling constraints both in joint space and in task space, and proposed an inverse kinematics method to re-use the learned skill in new situations. We demonstrated through experiments performed on two *Katana* robots that the approach could be applied successfully to learn new manipulation skills at a trajectory level, by generalizing over several demonstrations and by extending the learned tasks to new positions of objects. We finally discussed the advantages of Gaussian Mixture Model and Gaussian Mixture Regression for encoding, recognition, reproduction and evaluation issues.

The proposed approach presents advantages over our previous attempts at combining several constraints encoded in different data spaces. Compared to the geometric inverse kinematics approach used in [2], the approach proposed here can be extended to different robot architectures. Compared to the use of Lagrange optimization to find a metric of imitation performance [1], the proposed direct computation method does not require to analytically derive the cost function, and thus remains generic and statistically correct. Moreover, this direct computation method allows to compute the resulting constraints for the final controller in the form of a covariance matrix.

Further work goes toward learning the intrinsic dynamics of the motion. In the experiments presented here, time was considered as an explicit variable to encode the motion. We investigate methods to depart from the explicit representation of time by encapsulating the intrinsic dynamics of the motion in the GMM (e.g., by encapsulating position and velocity in GMM and by retrieving recursively through GMR a velocity command, given the current position) [34, 35]. This ongoing work aims at extending the proposed framework to motion with more complex dynamics (including sequential and cyclic behaviors), and to controllers that can adapt robustly to various external perturbations. Such encoding would also allow the user to provide partial demonstrations to the robot, which would increase the flexibility of the teaching process.

REFERENCES

[1] S. Calinon, F. Guenter, and A. Billard, "On learning, representing and generalizing a task in a humanoid robot," *IEEE Trans. on Systems, Man and Cybernetics, Part B*, vol. 37, no. 2, pp. 286–298, 2007.

[2] S. Calinon and A. Billard, "What is the teacher's role in robot programming by demonstration? - Toward benchmarks for improved learning," *Interaction Studies*, vol. 8, no. 3, pp. 441–464, 2007.

[3] A. Billard, Y. Epars, S. Calinon, G. Cheng, and S. Schaal, "Discovering optimal imitation strategies," *Robotics and Autonomous Systems*, vol. 47, no. 2-3, pp. 69–77, 2004.

[4] C. Nehaniv and K. Dautenhahn, "Of hummingbirds and helicopters: An algebraic framework for interdisciplinary studies of imitation and its applications," in *Interdisciplinary Approaches to Robot Learning*, ser. Robotics and Intelligent Systems, J. Demiris and A. Birk, Eds. World Scientific Press, 2000, vol. 24, pp. 136–161.

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

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

[7] A. Ude, "Trajectory generation from noisy positions of object features for teaching robot paths," *Robotics and Autonomous Systems*, vol. 11, no. 2, pp. 113–127, 1993.

[8] T. Inamura, I. Toshima, and Y. Nakamura, "Acquiring motion elements for bidirectional computation of motion recognition and generation," in *Experimental Robotics VIII*, B. Siciliano and P. Dario, Eds. Springer-Verlag, 2003, vol. 5, pp. 372–381.

[9] N. Delson and H. West, "Robot programming by human demonstration: Adaptation and inconsistency in constrained motion," in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, 1996, pp. 30–36.

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

[11] S. Vijayakumar and S. Schaal, "Locally weighted projection regression: An O(n) algorithm for incremental real time learning in high dimensional spaces," in *Proc. Intl Conf. on Machine Learning (ICML)*, 2000, pp. 288–293.

[12] D. Grimes, R. Chalodhorn, and R. Rao, “Dynamic imitation in a humanoid robot through non-parametric probabilistic inference,” in *Proc. Robotics: Science and Systems (RSS)*, Philadelphia, Pennsylvania, USA, August 2006.

[13] Z. Ghahramani and M. Jordan, “Supervised learning from incomplete data via an EM approach,” in *Advances in Neural Information Processing Systems*, J. D. Cowan, G. Tesauro, and J. Alspector, Eds., vol. 6. Morgan Kaufmann Publishers, Inc., 1994, pp. 120–127.

[14] S. Chiaverini, G. Oriolo, and I. Walker, “Kinematically redundant manipulators,” in *Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer, 2008, pp. 245–268.

[15] K. Grochow, S. Martin, A. Hertzmann, and Z. Popovic, “Style-based inverse kinematics,” in *Proc. ACM Intl Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH)*, Los Angeles, CA, USA, 2004, pp. 522–531.

[16] K. B. Petersen and M. S. Pedersen, “The matrix cookbook,” Technical Report. University of Denmark, 2008.

[17] J. Butcher, *Numerical Methods for Ordinary Differential Equations.* Chichester, UK: Wiley, 2003.

[18] M. Hersch and A. Billard, “Reaching with multi-referential dynamical systems,” *Autonomous robots*, vol. 25, pp. 71–83, 2008.

[19] D. Whitney, “The mathematics of coordinated control of prosthetic arms and manipulators,” *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 94, no. 4, pp. 303–309, 1972.

[20] S. Calinon, *Robot Programming by Demonstration: A Probabilistic Approach.* EPFL/CRC Press, 2009, ePFL Press ISBN 978-2-940222-31-5, CRC Press ISBN 978-1-4398-0867-2.

[21] D. Lee and Y. Nakamura, “Mimesis scheme using a monocular vision system on a humanoid robot,” in *Proc. IEEE Intl Conf. on Robotics and Automation (ICRA)*, Roma, Italy, April 2007, pp. 2162–2168.

[22] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, “Robot programming by demonstration,” in *Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Secaucus, NJ, USA: Springer, 2008, pp. 1371–1394.

[23] W. Cleveland, “Robust locally weighted regression and smoothing scatterplots,” *American Statistical Association*, vol. 74, no. 368, pp. 829–836, 1979.

[24] C. Atkeson, “Using local models to control movement,” in *Advances in Neural Information Processing Systems (NIPS)*, vol. 2, 1990, pp. 316–323.

[25] A. Moore, “Fast, robust adaptive control by learning only forward models,” in *Advances in Neural Information Processing Systems (NIPS)*, vol. 4. San Francisco, CA, USA: Morgan Kaufmann, 1992, pp. 571–579.

[26] S. Calinon and A. Billard, "Incremental learning of gestures by imitation in a humanoid robot," in *Proc. ACM/IEEE Intl Conf. on Human-Robot Interaction (HRI)*, Arlington, VA, USA, March 2007, pp. 255–262.

[27] D. Scott, "Multivariate density estimation and visualization," in *Handbook of Computational Statistics: Concepts and Methods*, W. H. J.E. Gentle and Y. Mori, Eds. Berlin, Germany: Springer-Verlag, 2004, pp. 517–538.

[28] H. Wold, "Estimation of principal components and related models by iterative least squares," in *Multivariate Analysis*, P. Krishnaiaah, Ed. New York, USA: Academic Press, 1966, pp. 391–420.

[29] S. Calinon and A. Billard, "Recognition and reproduction of gestures using a probabilistic framework combining PCA, ICA and HMM," in *Proc. Intl Conf. on Machine Learning (ICML)*, Bonn, Germany, August 2005, pp. 105–112.

[30] C. Rasmussen and C. Williams, *Gaussian processes for machine learning.* Cambridge, MA, USA: MIT Press, 2006.

[31] A. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

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

[33] P. Boyle and M. Frean, "Dependent gaussian processes," in *Advances in Neural Information Processing Systems*, vol. 17. MIT Press, 2005, pp. 217–224.

[34] M. Hersch, F. Guenter, S. Calinon, and A. Billard, "Dynamical system modulation for robot learning via kinesthetic demonstrations," *IEEE Trans. on Robotics*, vol. 24, no. 6, pp. 1463–1467, 2008.

[35] S. Calinon, P. Evrard, E. Gribovskaya, A. Billard, and A. Kheddar, "Learning collaborative manipulation tasks by demonstration using a haptic interface," in *Proc. Intl Conf. on Advanced Robotics (ICAR)*, Munich, Germany, June 2009.

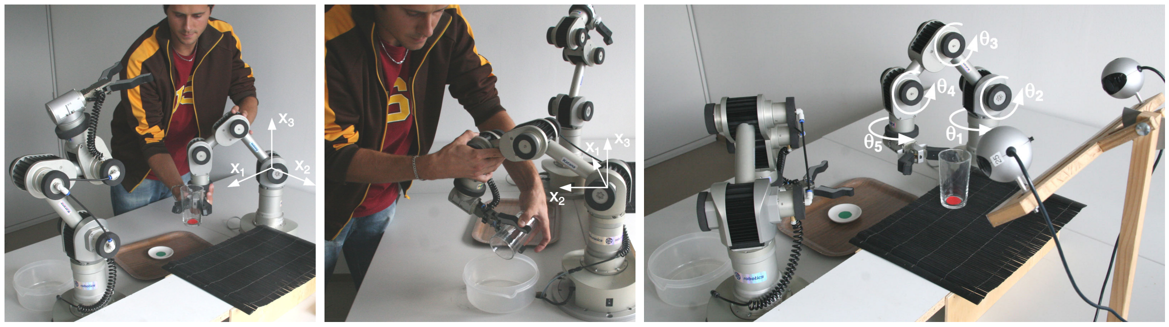

Figure 1: Kinesthetic demonstrations of the two tasks considered in this experiment, namely grasping and placing a glass on a coaster (*left*), and grasping and emptying a glass (*middle*). *Right:* Reproduction of the skill by the two robots, during which the initial positions of the objects are tracked by a stereoscopic vision system.

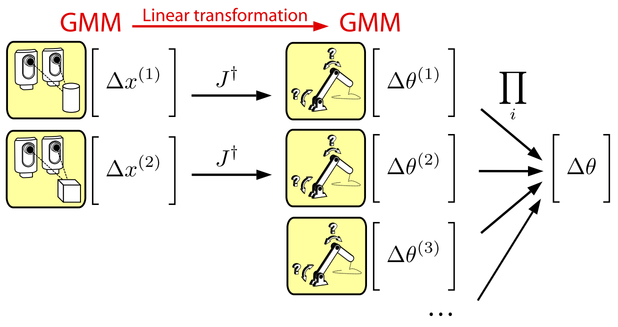

Figure 2: Illustration of the process used to retrieve a skill by considering constraints on various objects in task space (first two rows) as well as constraints in joint space (last row). The pseudoinverse Jacobian matrix $J^{\dagger}$ is employed to locally project the GMM representation of the constraints in task space to a corresponding representation in joint space. With the GMMs projected in joint space, an optimal solution can then be estimated through GMR by using the product properties of Gaussian distributions.

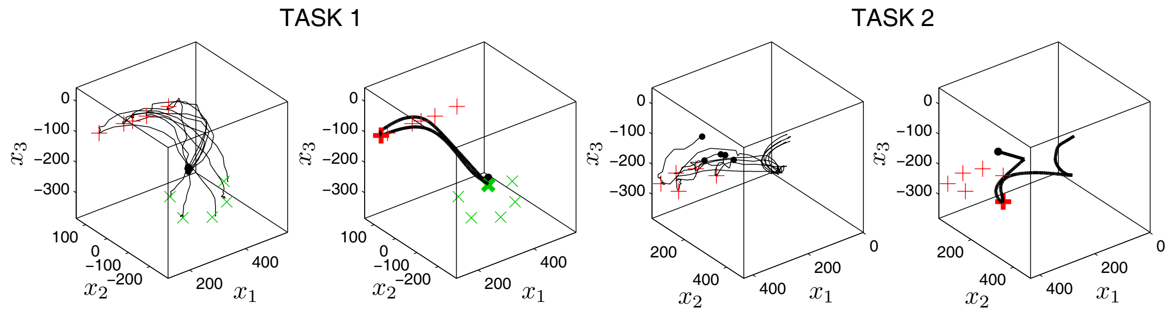

Figure 3: Demonstrations and reproduction in 3D Cartesian space for the two tasks. For each task, five demonstrations starting from different initial positions are considered (*first and third graphs*), where the trajectories are represented in the robot's frame of reference (see Figure 1), and the dots indicate the beginning of the motions. For the first task, the initial positions of the glass placed on the shelf are represented with '+' signs. The initial positions of the coaster on the table are represented with 'x' signs. For the second task, the initial positions of the glass (covering the coaster) are represented with '+' signs. The second and fourth graph show reproduction attempts for new situations (bold '+' and 'x' signs), by combining constraints in joint space and task space.

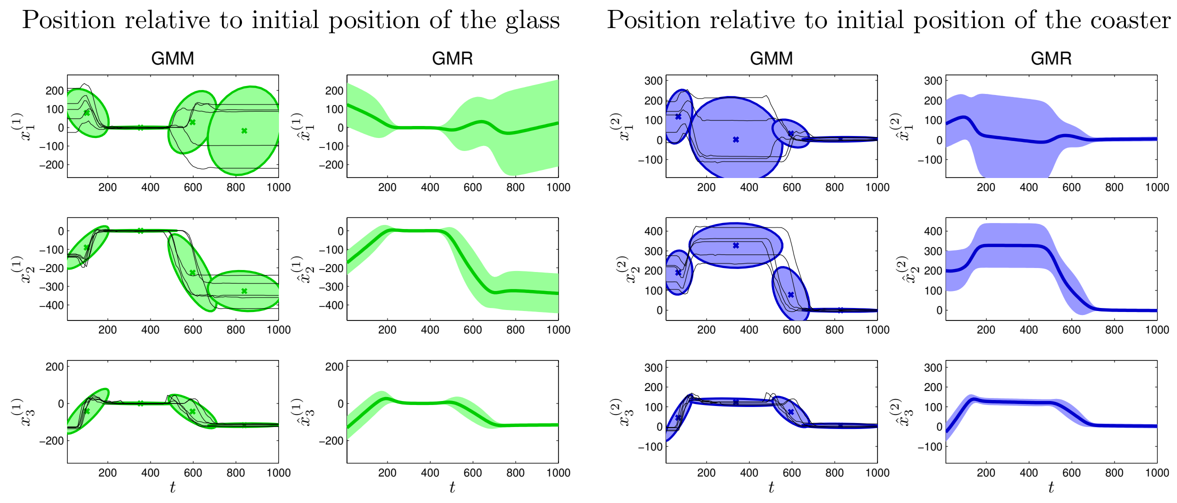

Figure 4: Automatic extraction of the constraints in task space for *TASK 1* (the corresponding frames of reference are depicted in Figure 1), where the two sets of graphs represent the constraints relative to the initial positions of the objects. GMMs with 4 Gaussians are used to encode the trajectories (selected by *Bayesian Information Criterion* (BIC) for each representation). The trajectories relative to the glass appear to be highly constrained between time steps 200 and 500 when reaching for the glass (*left*). The trajectories relative to the coaster are highly constrained at the end of the motion when placing the glass on the it (*right*).

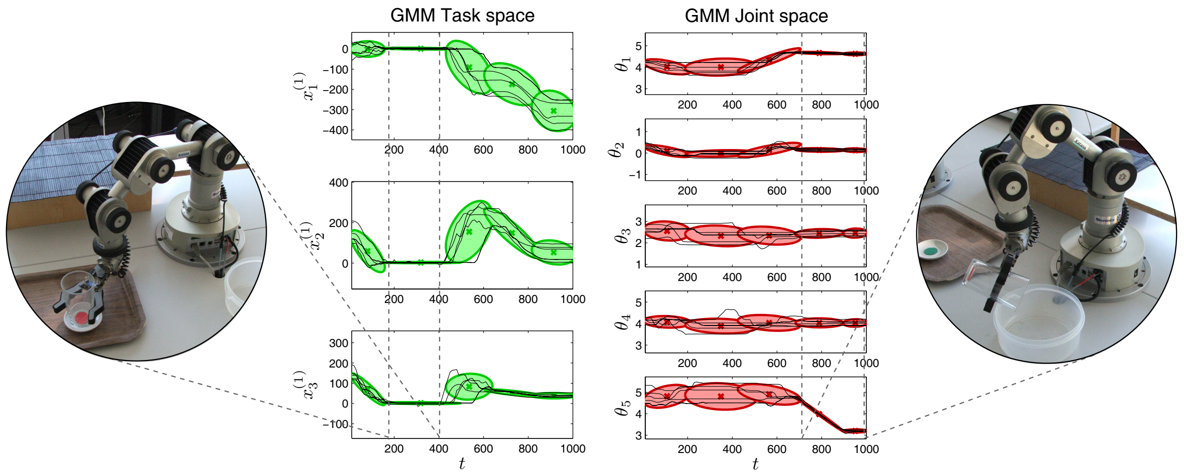

Figure 5: Automatic extraction of the task constraints for *TASK 2*, where GMMs with 5 Gaussians are selected by BIC to efficiently encode the skill. We see that the trajectories relative to the glass are highly constrained between time steps 200 and 400 (when reaching for the glass). Subsequently, the trajectories in joint space become more constrained. This occurs at the end of the motion, when emptying the glass in the basin by using a specific gesture. The snapshots illustrate a reproduction attempt by automatically selecting a controller that smoothly reproduces the extracted constraints.

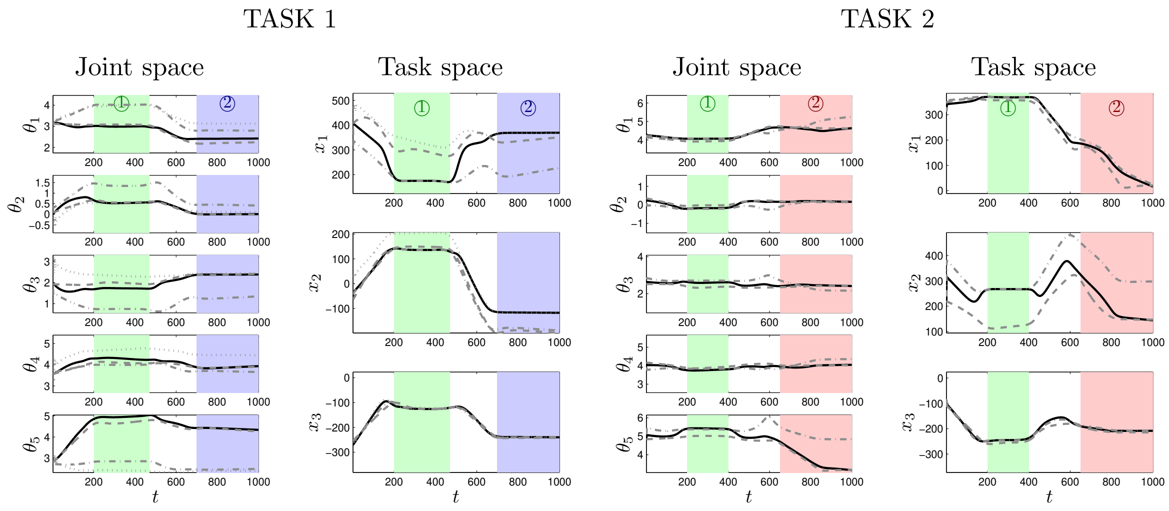

Figure 6: Reproduction attempts for *TASK 1* (*left*) and TASK 2 (*right*) by using the extracted constraints either independently or simultaneously. The trajectories represented by *solid line* show the final reproduction attempt by considering the constraints in task space and in joint space simultaneously. The trajectories represented by *dash-dotted line* consider only constraints for the first object in task space. The ones in *dotted line* consider only constraints for the second object in task space. Finally, the ones represented by *dashed line* take into account only constraints in joint space. For TASK 1, ① and ② correspond respectively to the time at which the robot grasps the glass and that at which it discards it on the coaster. For TASK 2, ① and ② correspond respectively to the time at which the robot grasps the glass and that when it empties the glass by an appropriate tilting.