Learning adaptive movements from demonstration and self-guided exploration

Danilo Bruno
Department of Advanced Robotics
Istituto Italiano di Tecnologia (IIT)
Via Morego, 30 - 16163 Genova - Italy

Sylvain Calinon
Idiap Research Institute
Rue Marconi 19, CH-1920
Martigny - Switzerland

Darwin G. Caldwell
Department of Advanced Robotics
Istituto Italiano di Tecnologia (IIT)
Via Morego, 30 - 16163 Genova - Italy

***Abstract*—The combination of imitation and exploration strategies is used in this paper to transfer sensory-motor skills to robotic platforms. The aim is to be able to learn very different tasks with good generalization capabilities and starting from a few demonstrations. This goal is achieved by learning a task-parameterized model from demonstrations where a teacher shows the task corresponding to different possible values of preassigned parameters. In this manner, new reproductions can be generated for new situations by assigning new values to the parameters, thus achieving very precise generalization capabilities. In this paper we propose a novel algorithm that is able to learn the model together with its dependence from the task-parameters, without specifying a predefined relationship or structure. The algorithm is able to learn the model starting from a few demonstrations by applying an exploration strategy that refines the learnt model autonomously. The algorithm is tested on a reaching task performed with a Barrett WAM manipulator.**

I. INTRODUCTION

The aim of this paper is to implement an algorithm that allows the transfer of sensorimotor skills to robotic platforms in a fast and efficient manner, with robust generalization capabilities. Usually humans develop these capabilities with an incremental learning process, mostly based on the exploration of many possible solutions [1]. This process is usually performed by children with a random and playful attitude [2], which allows them to discover very articulated solutions, but the overall process can be too slow and time consuming for robotics applications. A possible shortening of the process can be obtained by demonstrating the task to the robot and extracting the necessary information to restrict the exploration process. In this case, the exploration strategy can be coupled with a Learning by Demonstration strategy to allow the robot to generalize the skill and reproduce the learnt task even when the conditions change significantly with respect to the demonstrations [3], [4].

One way of coping with generalization is to use a task-parameterized model during the learning process. These models contain two different kinds of parameters, called model parameters and task parameters. The first ones are common to all the demonstrations and are inferred during the learning phase by using a single inference process involving all the demonstrations. The second ones are specific to each demonstration and distinguish each task instance from the other; for example they can be the position of the end-effector of the robot in task space, measured interaction forces with the environment, positions of external objects and so on.

After learning, the task can be reproduced in different situations by setting new values for task parameters, thus providing the envisaged generalization.

Task parameterized models can be decomposed into two broad categories. The first assumes a known relationship between the task and the parameters [5], [6], such as linear transformations between two frames of reference. These models are useful when the structure or task dependence is manifest. They are usually quite efficient, since they need a low number of demonstrations and have a good extrapolation capability.

The other category assumes that the dependence of the task on the parameters is unknown [7], [8], [9]. These models can usually be applied to a wider range of situations. This comes at the expense of a lower extrapolation capability, since the dependence is learnt from the demonstrations; moreover, a higher number of demonstrations is usually needed to build an accurate model. If a sufficient number of demonstrations is given, they usually work very well within the range of observed parameters.

In this paper we propose a statistical task-parameterized model that is able to learn the task together with the its relationship with the task parameters. The challenge is placed in keeping the number of demonstrations low, while still allowing the system to build a significant statistical model for new values of task-parameters that are far from the demonstrations.

The idea is to let the robot autonomously refine the model to gain a better precision during the reproduction of the task.

Instead of exploring the space of model parameters to optimize some global cost function, it is proposed to explore the space of task parameters to add new statistical points to be used in the inference process. After several iterations, the model parameters are updated based on the new samples, thus increasing the accuracy of the statistical model as a result of the addition of relevant datapoints during the exploration phase.

The statistical model fits the joint distribution of trajectory models and task parameters. A trajectory model is created

This work was partially supported by the STIFF-FLOP European project under contract FP7-ICT-287728.

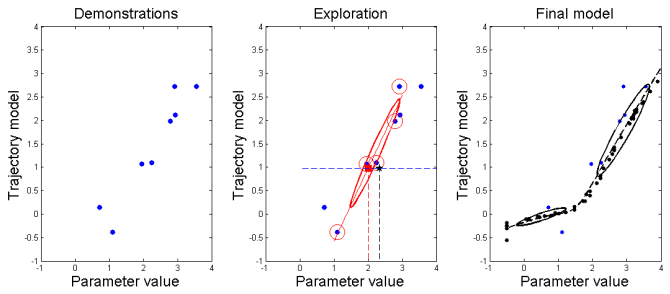

Fig. 1. The three steps of the algorithm explained. Left: demonstrations are given and represented as points in an abstract space. Center: the red Gaussian is built out of the points that are nearer to the desired parameter and a linear regression technique is used to estimate the trajectory (red square). The black star represents the actual value of the parameter after the execution of the task. The first estimate is discarded and the correct point is added to the dataset. Right: after a sufficient number of points have been explored, a final GMM model is built. The dashed line represents the learnt dependence between task parameters and trajectory models.

by representing each demonstration as a single point in an abstract space, by using a tensorial Gaussian Mixture Model (Section II-C). The task-parameterized model of the task is fit in the extended space, obtained by associating to each point representing a trajectory the corresponding value of the task parameters as additional dimensions.

During the exploration phase (Section II-D), an elitist regression technique is used to generate new trajectories for new values of the task parameters. For each of these new reproductions, the actual values of the task parameters is measured on the real system, which are concatenated to the trajectory model parameters and added to the previous set of points. As a matter of fact, the inferred trajectories are not considered as "wrong" whenever they do not result into the expected value of task parameters. They are instead considered as correct trajectories with wrong values of the parameters. This technique has links with random goal exploration [10] and can be seen as an algorithmic implementation of such strategy.

At the end of the exploration phase, a statistical model for the joint distribution of model parameters and task parameters is learnt and the points representing demonstrations and exploration steps are discarded (Section II-E). As a result, the robot is able to autonomously improve its ability, starting from a few demonstrations and ending up, independently from additional inputs from the user, with a usable statistical model that allows the retrieval of trajectories from new values of task parameters in a very fast manner and without external intervention. The advantage of the proposed technique is placed in the fact that the same code can be used to learn very different tasks.

II. PROBLEM SETTING

A. Notations

In the paper, lowercase bold letters ($\boldsymbol{t}$) denote vectors, uppercase bold letters ($\boldsymbol{X}$) represent matrices and calligraphic letters ($\mathcal{X}$) stand for tensors of higher order. When referring to components, parentheses are used, so that $\boldsymbol{t}^{(i)}$ denotes the i-th component of $\boldsymbol{t}$ and so on. The square brackets are used to denote sub-matrices and sub-vectors, e.g., $\boldsymbol{X} = [\boldsymbol{X}^{[t]^\top}, \boldsymbol{X}^{[p]^\top}]^\top$. Indices without parentheses are used to denote elements within a collection of objects. Finally, dots in the place of indices represent the full range of the given dimension (fiber) (e.g. $\mathcal{X}^{(\bullet,\bullet,j)} = \boldsymbol{X}^j$ represents a collection of matrices obtained by fixing an index of a third order tensor).

B. General overview of the method

The task-parameterized model is a statistical model representing the joint distribution of trajectory model parameters and task parameters. Each datapoint consists of a vector $\boldsymbol{x} = [\boldsymbol{x}^{[t]^\top}, \boldsymbol{x}^{[p]^\top}]^\top$, where the vector $\boldsymbol{x}^{[t]}$ is the model of a demonstrated trajectory and $\boldsymbol{x}^{[p]}$ is the value of the task parameter associated to it. We call $\boldsymbol{x}$ the task vector. The algorithm consists of three consecutive steps (Fig. 1), that are detailed in the rest of the section:

1. a model of the demonstrated trajectories in a vector form is calculated and augmented with the corresponding task parameters (II-C);
2. given new values of the task parameters $\boldsymbol{x}^{[p]}$, new trajectory model parameters $\boldsymbol{x}^{[t]}$ are calculated by using an elitist linear regression technique; the new trajectory is reproduced on the robot, the actual value $\boldsymbol{x}_*^{[p]}$ of the task parameter is measured and a new task vector $\boldsymbol{x}_* = [\boldsymbol{x}^{[t]^\top}, \boldsymbol{x}_*^{[p]^\top}]^\top$ is added to the previous points (II-D);
3. a statistical model is used to encode all the task vectors $\boldsymbol{x}$, which can then be discarded; the resulting model can be used to infer new trajectory models for new values of task parameters (II-E).

C. Trajectory model

Each demonstration is encoded separately from the others (one for each task parameter value) by using Gaussian Mixture Models (GMM), through a DS-GMR dynamical system formulation (see [11] for details). In short, the trajectory is generated by a virtual spring-damper system; for each point $\boldsymbol{a}_n$ of the trajectory, a virtual attractor $\boldsymbol{y}_n$ is calculated with

$$\ddot{\boldsymbol{a}}_n = -\boldsymbol{K}_P\,(\boldsymbol{a}_n - \boldsymbol{y}_n) - \boldsymbol{K}_V\,\dot{\boldsymbol{a}}_n \quad . \tag{1}$$

The trajectory of the attractor, consisting of the datapoints $\boldsymbol{\xi}_n = [t_n, \boldsymbol{y}_n^\top]^\top$, is encoded into a statistical model in the form of a Gaussian mixture model (GMM). The positions of the attractor can be retrieved at each time step by using Gaussian mixture regression (GMR) [12], using time as input. The original trajectory can be reconstructed from the trajectory of the attractors by using Eq. (1). This encoding of the trajectory is stable to perturbations and can easily generate the correct path, even if the starting position is not the same as the demonstrated one.

In order to make the model of each trajectory comparable with the others, we must be made up of the same number of components and that the components are time aligned. In order to perform both operations automatically, the demonstrations are stored into a third order tensor $\mathcal{X}$. In this representation, each matrix $\mathcal{X}^{(\bullet,\bullet,j)}$, obtained by fixing the third index, stores

a demonstration, consisting of a matrix, whose columns are the datapoints $\boldsymbol{\xi}_n$. The tensor $\mathcal{X}$ has dimensions (D, M, P), where D is the dimension of each datapoint, M is the number of datapoints for each demonstration and P is the number of demonstrations. Typically, D is of the order of $3-10$, M of 200 and P of 10.

The dataset is encoded into a tensorial Gaussian Mixture Model, by using a tensorial normal distribution representation [13]. The distribution is fit over the matrices $\boldsymbol{X}_n = \mathcal{X}^{(\bullet,n,\bullet)}$, representing the collection of datapoints for all demonstrations at time step n.

The probability that each sample $\boldsymbol{X}_n$ is generated by the envisaged model is

$$\mathcal{P}(\boldsymbol{X}_n) = \sum_{i=1}^{K} \pi_i \, \mathcal{N}(\boldsymbol{X}_n | \boldsymbol{M}_i, \mathcal{S}_i) \quad , \tag{2}$$

where $\boldsymbol{M}_i \in \mathbb{R}^{D\times P}$ is a second order tensor representing the mean of the distribution, $\mathcal{S}_i \in \mathbb{R}^{D\times P\times P\times D}$ is a fourth order tensor encapsulating covariance information, and K is the number of components of the GMM.

The tensor normal distribution is described as

$$\mathcal{N}(\boldsymbol{X}|\boldsymbol{M},\mathcal{S}) = \sqrt{\frac{|\mathcal{S}|^{-1}}{(2\pi)^{DP}}} \exp\left[-\frac{1}{2}\sum_{ijrs} \boldsymbol{Z}^{\top(ij)} (\mathcal{S}^{-1})^{(ijrs)} \boldsymbol{Z}^{(rs)}\right],$$

where $\boldsymbol{Z} = \boldsymbol{X} - \boldsymbol{M}$. For the details the reader is referred to [13].

If we make the hypothesis that the information of each demonstration is not correlated to the others, we can then restrict the covariance tensor to be diagonal in the indices running over demonstrations, i.e.,

$$\begin{aligned} &\mathcal{S}^{(ikkj)} = \boldsymbol{\Sigma}^{(ij)} \;\; \forall i,j \in \{1\dots D\} \text{ and } \forall k \in \{1\dots P\} \;, \\ &\mathcal{S}^{(ikrj)} = 0 \quad \forall i,j \in \{1\dots D\} \text{ and } k \neq r \in \{1\dots P\}. \end{aligned} \tag{3}$$

In this case, the normal distribution can be written (up to normalization) as

$$\begin{aligned} \mathcal{N}(\boldsymbol{X}|\boldsymbol{M},\mathcal{S}) &\propto \exp\left(-\frac{1}{2}\sum_{ijrs} \boldsymbol{Z}^{\top(ij)} (\mathcal{S}^{-1})^{(ijrs)} \boldsymbol{Z}^{(rs)}\right) = \\ &= \prod_{j=1}^{P} \exp\left(-\frac{1}{2}(\boldsymbol{z}^j)^\top (\boldsymbol{\Sigma}^{-1})^j \boldsymbol{z}^j\right) \propto \prod_{j=1}^{P} \mathcal{N}(\boldsymbol{\xi}^j | \boldsymbol{\mu}^j, \boldsymbol{\Sigma}^j), \end{aligned} \tag{4}$$

where $\boldsymbol{z}^j = \boldsymbol{\xi}^j - \boldsymbol{\mu}^j = \mathcal{Z}^{(\bullet,j)}$ and $\boldsymbol{\Sigma}^j = \mathcal{S}^{(\bullet,j,j,\bullet)}$.

Each component of Eq. (2) can be rewritten according to Eq. (4), resulting in

$$\mathcal{P}(\boldsymbol{X}_n) = \sum_{i=1}^{K} \pi_i \prod_{j=1}^{P} \mathcal{N}(\boldsymbol{\xi}_n^j | \boldsymbol{\mu}_i^j, \boldsymbol{\Sigma}_i^j) \quad , \tag{5}$$

where $\boldsymbol{\mu}_i^j$ and $\boldsymbol{\Sigma}_i^j$ represent the mean and the covariance matrix corresponding to the component i and demonstration j, and $\boldsymbol{x}_n^j$ is the n-th datapoint of demonstration j. All the GMMs share the same priors π_i, as envisaged.

Taking the M datapoints into account and writing the probability explicitly, we obtain that the likelihood for the whole dataset is

$$\mathcal{P}(\mathcal{X}) = \prod_{n=1}^{M} \sum_{i=1}^{K} \pi_i \prod_{j=1}^{P} \mathcal{N}(\boldsymbol{\xi}_n^j | \boldsymbol{\mu}_i^j, \boldsymbol{\Sigma}_i^j) \;. \tag{6}$$

The model parameters can be estimated by maximizing the log-likelihood of the dataset with respect to $\boldsymbol{\mu}_i^j$ and $\boldsymbol{\Sigma}_i^j$. Skipping all the calculations, we have that

$$\boldsymbol{\mu}_i^j = \frac{\sum_n \gamma_{n,i} \boldsymbol{\xi}_n^j}{\sum_n \gamma_{n,j}}, \tag{7}$$

$$\boldsymbol{\Sigma}_i^j = \frac{\sum_n \gamma_{n,i} (\boldsymbol{\xi}_n^j - \boldsymbol{\mu}_i^j)(\boldsymbol{\xi}_n^j - \boldsymbol{\mu}_i^j)^\top}{\sum_n \gamma_{n,i}}, \tag{8}$$

$$\pi_i = \frac{\sum_{n=1}^{M} \gamma_{n,i}}{M} \tag{9}$$

where

$$\gamma_{n,i} = \frac{\pi_i \prod_{j=1}^{P} \mathcal{N}\left(\boldsymbol{\xi}_n^j | \boldsymbol{\mu}_i^j, \boldsymbol{\Sigma}_i^j\right)}{\sum_{k=1}^{K} \pi_k \prod_{j=1}^{P} \mathcal{N}\left(\boldsymbol{\xi}_n^j | \boldsymbol{\mu}_k^j, \boldsymbol{\Sigma}_k^j\right)}. \tag{10}$$

The parameters encode a different trajectory for each value of j. The resulting GMM is then turned into a single point in an abstract trajectory model space. In this paper, we store the mean vector $\boldsymbol{\mu}_i^j$ and the principal vector $\boldsymbol{w}_i^j$ of the covariance matrix of every component of the GMM into the vector

$$\boldsymbol{x}_j^{[t]} = [\boldsymbol{\mu}_1^{j\top}, \boldsymbol{w}_1^{j\top}, \dots, \boldsymbol{\mu}_K^{j\top}, \boldsymbol{w}_K^{j\top}]^\top \tag{11}$$

of dimension $N_t = 2KD$, where K is the number of components of the tensor GMM. Each trajectory model vector $\boldsymbol{x}_j^{[t]}$ is augmented with the N_p-dimensional vector $\boldsymbol{x}_j^{[p]}$, containing the value of the task parameters associated with the demonstration. The resulting task vector $\boldsymbol{x}_j = [\boldsymbol{x}_j^{[t]\top}, \boldsymbol{x}_j^{[p]\top}]^\top$ represents all the information stored with each demonstration. We denote by $\mathcal{T} = \{\boldsymbol{x} | \boldsymbol{x} = [\boldsymbol{x}^{[t]\top}, \boldsymbol{x}^{[p]\top}]^\top\}$ the space of task instances. The priors π_i are common to all trajectories and are stored apart since they do not change.

D. The exploration process

A trajectory for a new value of the task parameters can be obtained by using an inference technique, where each trajectory model and associated task parameters are represented in the form of output variables (trajectory model $\boldsymbol{x}^{[t]}$) and query points (task parameter $\boldsymbol{x}^{[p]}$).

In this paper we follow an elitist approach: each time a trajectory for a new task parameter value $\boldsymbol{x}^{[p]}$ is required, the set $I_{best} \subset \{1\dots P\}$, composed by the $N = P/2$ points whose parameter value is closest to the enquired one, is selected. A Normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is fit to the selected points, with mean and covariance represented as

$$\boldsymbol{\mu} = \frac{1}{N} \sum_{i \in I_{best}} \boldsymbol{x}_i \quad , \quad \boldsymbol{\Sigma} = \frac{1}{N} \sum_{i \in I_{best}} (\boldsymbol{x}_i - \boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu})^\top + \boldsymbol{\Sigma}_0,$$

with

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}^{[t]} \\ \boldsymbol{\mu}^{[p]} \end{bmatrix} \quad , \quad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}^{[t]} & \boldsymbol{\Sigma}^{[tp]} \\ \boldsymbol{\Sigma}^{[pt]} & \boldsymbol{\Sigma}^{[p]} \end{bmatrix}, \tag{12}$$

where $\boldsymbol{\Sigma}_0$ is a small regularization term.

The best estimate for the trajectory model corresponding to the desired task parameter $\boldsymbol{x}^{[p]}$ is inferred by regression as the mean for the conditional Gaussian $\mathcal{N}(\boldsymbol{x}^{[t]}|\boldsymbol{x}^{[p]})$, i.e.

$$\boldsymbol{x}^{[t]} = (\boldsymbol{\mu}^{[t]}|\boldsymbol{x}^{[p]}) = \boldsymbol{\mu}^{[t]} + \boldsymbol{\Sigma}^{[tp]}(\boldsymbol{\Sigma}^{[p]})^{-1}(\boldsymbol{x}^{[p]} - \boldsymbol{\mu}^{[p]}). \tag{13}$$

The GMM encoding the trajectory attractors is reconstructed back from the inferred point and the resulting trajectory is run on the robot (or in simulation). In this paper the covariance matrix is reconstructed by adding to the principal vector a small fixed regularization term. The real values $\boldsymbol{x}_*^{[p]}$ of the task parameters achieved by the extracted trajectory are measured and the point $\boldsymbol{x} = [\boldsymbol{x}^{[t]^\top}, \boldsymbol{x}_*^{[p]^\top}]^\top$ is added to the current dataset.

This exploration strategy differs from the one commonly used in Reinforcement Learning, since it is not directly driven towards an improvement of the policy by maximizing a cost function. Instead, it is aimed at exploring the task parameters space to increase the number of experimental points in a region and, consequently, the accuracy of the model. In this paper, the new task parameters are sampled from the distribution of the demonstrated task parameters, that is fit over the values of task parameters associated to the demonstrations.

An interesting aspect of this exploration strategy is to couple it with active learning techniques, in order to determine what are the areas of the space of task parameters where it is necessary to sample to speed up learning. This strategy can be also exploited to explore areas where no demonstration was given, in order to improve the extrapolation capabilities of the model. A discussion about this topic can be found in Section IV.

E. The task-parameterized model

A statistical model of the task is built by using a Gaussian Mixture Model (GMM) in the space $\mathcal{T}$. If only one component is used, the model corresponds to a linear regression problem; if the dependence on the parameters is non-linear, a higher number of components can be used to approximate the non-linear behaviour with a locally linear one, provided a sufficient number of components is used. The number of components can be eventually learnt by using non-parametric statistical models [14]. The posterior probability distribution of the statistical model in the space $\mathcal{T}$ is

$$\mathcal{P}(\boldsymbol{x}) = \sum_{k=1}^{K^*} \pi_k^* N(\boldsymbol{x}|\boldsymbol{\mu}_k^*, \boldsymbol{\Sigma}_k^*), \tag{14}$$

where $\pi_i^*, \boldsymbol{\mu}_k^*, \boldsymbol{\Sigma}_k^*$ are the model parameters, $\boldsymbol{x} \in \mathcal{T}$ is a datapoint encoding a task instance and K^* is the total number of components. The model parameters $\pi_k^*, \boldsymbol{\mu}_k^*$ and $\boldsymbol{\Sigma}_k^*$ can be obtained by using a standard EM algorithm.

New trajectories can be inferred from a new task parameters value $\boldsymbol{x}^{[p]}$ by using Gaussian Mixture Regression (GMR) [12], which is fast and reliable in the reproduction phase (it can be used online). If the model parameters are represented as

$$\boldsymbol{\mu}_i^* = \begin{bmatrix} \boldsymbol{\mu}_i^{*[t]} \\ \boldsymbol{\mu}_i^{*[p]} \end{bmatrix} \quad , \quad \boldsymbol{\Sigma}_i^* = \begin{bmatrix} \boldsymbol{\Sigma}_i^{*[t]} & \boldsymbol{\Sigma}_i^{*[tp]} \\ \boldsymbol{\Sigma}_i^{*[pt]} & \boldsymbol{\Sigma}_i^{*[p]} \end{bmatrix}, \tag{15}$$

a new value $\boldsymbol{x}^{[t]}$ of the trajectory model can be obtained from a parameter value $\boldsymbol{x}^{[p]}$ as

$$\boldsymbol{x}^{[t]} = \sum_{i=1}^{K^*} h_i \left[\boldsymbol{\mu}_i^{*[t]} + \boldsymbol{\Sigma}_i^{*[tp]}(\boldsymbol{\Sigma}_i^{*[p]})^{-1}(\boldsymbol{x}^{[p]} - \boldsymbol{\mu}_i^{*[p]}) \right], \tag{16}$$

with

$$h_i = \sum_{i=1}^{K^*} \pi_i^* \mathcal{N}(\boldsymbol{x}^{[p]}|\boldsymbol{\mu}_i^{*[p]}, \boldsymbol{\Sigma}_i^{*[p]}). \tag{17}$$

This approach works well within the explored task parameters space (as a matter of fact, any extrapolation outside the space of task parameter explored by the robot is linear).

The data are now discarded and we are left with a parametric probabilistic model of the task together with the parametric dependence. This means that the information can be stored in a compact parametric model, whence it can be fast retrieved during the reproduction phase.

III. EXPERIMENTS

In this section we provide 2 experiments to show how the method works. The algorithm is evaluated by the reproduction error across the different steps of the algorithm, measured as the distance between the predicted values of the task parameters and the actual ones. The measurement is performed by sampling new values of task parameters and evaluating the average error produced over a fixed number of trials. In the first step, only demonstrations are used to infer new trajectories by using Eq. (13); in the second step, exploration points are added before using Eq. (13); in the third step, the final GMM is fit in $\mathcal{T}$ and GMR used for reproduction. The results are averaged over a fixed number of runs of the whole procedure. The results are shown in Figures 2 and 3.

A. Reaching task

The first example is a 3D reaching task. Bell-shaped trajectories connecting a starting point P_0 to an ending point P_1 are demonstrated in 3D space. The parameters of the task are the starting and ending positions, resulting into a 6-dimensional task parameter space. The trajectory model is fit by a GMM with 4 components ($K = 4$), so the dimension of the trajectory model space is $N_t = 2KD = 32$, resulting into 38 dimensions for $\mathcal{T}$. A total number of $P = 12$ trajectories is demonstrated.

Exploration is performed by sampling new initial and final points from the distribution of demonstrated task parameters. The final retrieved trajectories are shown in Fig. 2.

As we can see in Fig. 2, the initial estimates of the trajectories using the demonstrations is quite poor (Step 1), since too few points are present in the 38-dimensional space where inference is performed. But we can see that during exploration (Step 2), the precision improves and reaches a small error once the final task-parameterized model is used (Step 3).

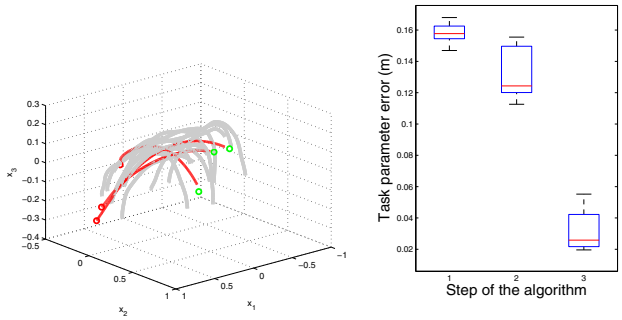

Fig. 2. *Left:* three final trajectories (in red) reproduced by using the task-parameterized model for new values of task parameters. The grey trajectories represent the demonstrations. *Right:* the error between the predicted and measured value of the position of the final point in the reproduced trajectory is used to evaluate the performance of the algorithm. Each boxplot corresponds to a step of the algorithm. The exploration step (2) consists of 50 trials. The results are averaged over 30 runs.

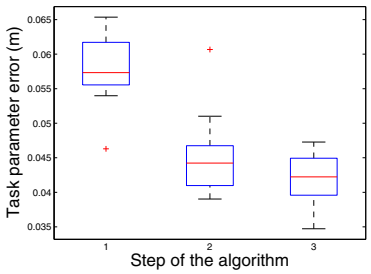

Fig. 3. The error between the predicted and measured value of the position of the final point in the reproduced trajectory is used to evaluate the performance of the algorithm. Each boxplot corresponds to a step of the algorithm. The exploration step (2) consists of 30 trials. The results are averaged over 30 runs.

B. Joint space learning with a 4DOF WAM Arm

A 4DOF WAM arm is used to perform a reaching task. The robot reaches a point in the 3D workspace placed either on its left or right side. An obstacle is put in front of it, separating the 2 areas, see Fig. 4. In order to avoid the obstacle with the whole body, during the demonstrations, the elbow of the robot is moved on the same side as the reaching point.

In this case, the task parameters are in Cartesian space (target positions of the tip of the manipulator), while the trajectory is encoded in joint space (the movement of the elbow is part of the task to learn). Thus, the unknown relation between the task parameters and the trajectory in joint space needs to be discovered by the learning algorithm. In this experiment, 10 demonstrations are given; the task parameter space is 3-dimensional, while the trajectories model in joint space are encoded by a 3-components GMM, resulting into $N_t = 2KD = 30$ trajectory model parameters and a 33 dimensional space $\mathcal{T}$.

During exploration, 30 trials are run in simulation with the Matlab Robotics Toolbox [15]. In this case, the choice of running the exploration in simulation is neither restrictive nor compulsory. Since the trajectory is executed on the robot by using a joint-space positional control, only the forward kinematics is needed to evaluate the final position of the tip of the robot from a given joint trajectory. So, the choice of running the exploration in simulation is faster and reliable. In order to validate the procedure, some trajectories were run on the robot and the same value of the final task parameter was measured.

The task-parameterized model consists of a 2 components GMM (the choice was here made heuristically by knowing that there were 2 different behaviours corresponding to the left and right points). The model is used to generate trajectories on the real robot. The reproduction error on the final position of the end effector is shown in Fig. 3, which decreases along the steps of the algorithm. The execution reliability is also improved, as we can observe from the disappearance of outliers in the last boxplot.

IV. DISCUSSION

The advantage of the proposed method is placed in its combination of imitation and exploration mechanisms and in its generality. Since no model of the dependence of the task from the parameters is given, the only inputs needed are the demonstrations of trajectories together with the corresponding task parameters. So, the same code can be used independently of the task and model encoding.

The encoding of each demonstrated trajectory into a single point by storing the mean vector and the principal vector of the covariance matrix for each component was an arbitrary choice, used as a trade-off between being precise in the generation of the trajectory and reducing the number of the parameters to improve the learning speed. The whole covariance matrix could be eventually used, but the reconstruction from new parameters would need a higher number of exploration steps to be performed correctly.

The exploration phase is the most interesting part of the process. In this paper, it is performed by sampling from the distribution of the task parameters that is learnt from the demonstrations to improve the precision of the reproductions, but other approaches are possible.

A link with active learning strategies can be established to allow the robot to autonomously choose what are the most interesting task parameters to explore. An interesting overview of the possible techniques can be found in [16]. Examples of promising strategies include samplings directed to areas showing the least exploration, to areas of highest reproduction errors, or to areas where the learning curve is the fastest. This paper thus has close links with exploration-driven developmental robotics [10] and intrinsic motivation search strategies [3]. Alternative choices can be given by a scaffolded exploration strategy, where an additional supervision of the user is added to guide the robot towards interesting unexplored areas [3].

The paper shares similarities with other task-parameterized approaches to robot learning. Stulp *et al.* in [8] use a learning from demonstration approach to build a task-parameterized

Fig. 4. Execution of the task on the WAM arm. The red line represents the trajectory of the end-effector.

version of Dynamic Movement Primitives (possessing additional task parameters that can be used to generalize trajectories). The paper uses a similar regression technique to handle the reproduction phase but does not provide an exploration strategy, preventing the approach to extrapolate outside the area of demonstrated task parameters.

Da Silva *et al.* in [17] uses an approach to exploration framed within Reinforcement Learning that is similar to the one we proposed, by adding the "wrong" trajectories to the demonstrations with the correct parameter value. The main difference is placed in the representation of the parameterized model, which uses a manifold approximator to encode the task/parameter relationship. This choice makes the model quite precise at reproduction but requires the encoding of different manifolds in different areas of the task parameters space, which is avoided in our approach by the use of the GMM.[1]

Finally, it can be underlined that the proposed approach can also be used with multivalued task-parameters relationships, where different trajectories correspond to the same task parameter. In this paper we make the hypothesis that a unimodal distribution is retrieved through the GMR procedure and that this is sufficient to describe the model-task dependence of the experiments. But if needed, the same approach could be used to retrieve a multimodal distribution.

V. CONCLUSION

We described a three step algorithm that learns a task-parametrized motion model together with the dependence of the motion from the task parameters. The algorithm reduces the level of supervision needed to learn the task, initialized by a set of demonstrations and the values of the corresponding task parameters (without needing to specify their actual nature or role within the task).

In order to reduce the number of demonstrations required to correctly learn the task, an exploration phase is exploited by the agent to explore the task-parameter space and improve the precision of the model. The algorithm was tested on a WAM arm manipulator in a reaching task.

REFERENCES

[1] J. Konczak and J. Dichgans, "The development towards stereotypic arm kinematics during reaching in the first 3 years of life," *Exp. Brain Res.*, vol. 117, pp. 346–354, 1997.

[1]This representation can also be useful in future works to exploit the covariance information to guide the exploration.

[2] L. Smith and M. Gasser, "The development of embodied cognition: six lessons from babies," *Artificial Life*, vol. 11, pp. 13–29, 2005.

[3] S. Nguyen and P.-Y. Oudeyer, "Properties for efficient demonstrations to a socially guided intrinsically motivated learner," in *21st IEEE International Symposium on Robot and Human Interactive Communication*, Paris, France, 2012.

[4] C. Moulin-Frier, S. Nguyen, and P. Oudeyer, "Self-organization of early vocal development in infants and machines: The role of intrinsic motivation." *Frontiers in Cognitive Science*, vol. 4, p. 1006, 2014.

[5] S. Calinon, T. Alizadeh, and D. G. Caldwell, "On improving the extrapolation capability of task-parameterized movement models," in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, Tokyo, Japan, November 2013, pp. 610–616.

[6] D. Herzog, A. Ude, and V. Krueger, "Motion imitation and recognition using parametric hidden Markov models," in *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*, Daejeon, Korea, 2008.

[7] T. Matsubara, S.-H. Hyon, and J. Morimoto, "Learning parametric dynamic movement primitives from multiple demonstrations," *Neural Networks*, vol. 24, no. 5, pp. 493–500, June 2011.

[8] F. Stulp, G. Raiola, A. Hoarau, S. Ivaldi, and O. Sigaud, "Learning compact parameterized skills with a single regression," in *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*, Tokyo, Japan, November 2013, pp. 417–422.

[9] A. Ude, A. Gams, T. Asfour, and J. Morimoto, "Task-specific generalization of discrete and periodic dynamic movement primitives," *IEEE Transactions on Robotics*, vol. 26, no. 5, pp. 800–815, 2010.

[10] C. Moulin-Frier and P. Oudeyer, "Exploration strategies in developmental robotics: a unified probabilistic approach," in *Proc. IEEE Intl. Conf. on Developmental and Leraning and Epigenetic Robotics (ICDL-Epirob)*, 2013.

[11] S. Calinon, Z. Li, T. Alizadeh, N. G. Tsagarakis, and D. G. Caldwell, "Statistical dynamical systems for skills acquisition in humanoids," in *Proc. IEEE Intl Conf. on Humanoid Robots (Humanoids)*, Osaka, Japan, 2012, pp. 323–329.

[12] Z. Ghahramani and M. I. 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.

[13] P. J. Basser and S. Pajevic, "Spectral decomposition of a 4th-order covariance tensor: Applications to diffusion tensor MRI," *Signal Processing*, vol. 87, pp. 220–236, 2007.

[14] D. Bruno, S. Calinon, and D. G. Caldwell, "Bayesian nonparametric multi-optima policy search in reinforcement learning," in *AAAI Conference on Artificial Intelligence*, Bellevue, Washington, USA, 2013.

[15] P. I. Corke, "Matlab toolboxes: robotics and vision for students and teachers," *IEEE Robotics and Automation Magazine*, vol. 14, no. 4, pp. 16–17, December 2007.

[16] P.-Y. Oudeyer and F. Kaplan, "What is intrinsic motivation? a typology of computational approaches," *Frontiers in Neurorobotics*, vol. 1, p. 6, 2007.

[17] B. da Silva, G. Baldassarre, G. Konidaris, and A. Barto, "Learning parameterized motor skills on a humanoid robot," in *Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA 2014).*, Hong Kong, China, 2014.