Accelerated Sensorimotor Learning of Compliant Movement Primitives

Tadej Petrič, *Member, IEEE*, Andrej Gams, *Member, IEEE*, Luca Colasanto, *Member, IEEE*, Auke J. Ijspeert, *Senior Member, IEEE*, and Aleš Ude, *Member, IEEE*

***Abstract*—Autonomous trajectory generation through generalization requires a database of motion, which can be difficult and time consuming to obtain. In this paper, we propose a method for autonomous expansion of a database for the generation of compliant and accurate motion, achieved through the framework of compliant movement primitives (CMPs). These combine task-specific kinematic and corresponding feed-forwarddynamic trajectories. The framework allows for generalization and modulation of dynamic behavior. Inspired by human sensorimotor learning abilities, we propose a novel method that can autonomously learn task-specific torque primitives (TPs) associated to given kinematic trajectories, encoded as dynamic movement primitives. The proposed algorithm is completely autonomous, and can be used to rapidly generate and expand the CMP database. Since CMPs are parameterized, statistical generalization can be used to obtain an initial TP estimate of a new CMP. Thereby, the learning rate of new CMPs can be significantly improved. The evaluation of the proposed approach on a Kuka LWR-4 robot performing a peg-in-hole task shows fast TP acquisition and accurate generalization estimates in real-world scenarios.**

***Index Terms*—Autonomous learning, compliant movement primitives (CMPs), internal dynamic models.**

I. Introduction

WHEN generating new motion trajectories, statistical generalization provides an option to synthesize a similar trajectory, appropriate for a new task within the training space, from a set of recorded movements [1].

However, two main challenges constrain the applicability of generalization.

1) The database needs to be provided beforehand, resulting in substantial initial programming effort for the user.
2) Generalization only works within the training space.

In this paper, we propose an algorithm for autonomous expansion of the database, where the robot uses iterative learning to optimize its behavior for new task conditions. The newly generated and optimized trajectories can either increase the density of the database within the training space, or expand it beyond its current limits to provide a greater area of applicability. Hence, minimal initial effort is left to the user. As a side note, in order to continuously transition among the trajectories in the database, the user has to ensure their similarity. As demonstrated in [1], the transition between trajectories originating from variations of the semantically equivalent task is smooth in real-world problems because it is unlikely that a completely different strategy would be used to solve a task in different but similar situations.

Learning of movement trajectories that are to expand the database can be a lengthy process, depending on the learning approach. For example, explorative methods, such as reinforcement learning, might take a long time because they need a high number of repetitions [2]. On the one hand, such methods were, for example, applied for learning of force profiles, which were combined with kinematic trajectories to accomplish in-contact tasks [3], [4]. Supervised methods, such as iterative learning control (ILC), on the other hand, are faster, but need a reference to optimize to [5]. Yet even these might take too long to generate a database that will allow statistical methods to generate an accurate trajectory for the given task parameters. In this paper, we show that we can considerably accelerate the learning by providing a better initial estimate, which originates from the at-the-time available database. Thus, iterative learning provides new entries into the database, while statistical generalization from the said database provides initial estimates for iterative learning. Consequently, the generalization results improve with every new entry, which also means that the initial approximations for explorative learning gradually improve as well.

The need for fast generation of new and at the same time accurate trajectories comes from the necessity to operate in unstructured environments, such as human everyday environments and homes. Operating in such environments and with humans requires that the robot is compliant in case of potential undesired contact with objects, and even more importantly, humans. Accurate dynamical models of the robot and the task allow for compliant and accurate behavior [6]. However, these models are difficult to obtain, so methods of learning task-specific models have been proposed [7]. One such method is with the use of Compliant Movement Primitives (CMPs) [7], which encode both the kinematic trajectory and the corresponding joint torques for the given task. The kinematic trajectory takes the form of a dynamic movement primitive (DMP) [8], retaining all the properties of modulation and adaptation of a DMP. The dynamic trajectory, i.e., the joint torques encoded as torque primitives (TPs), are given as a set of weighted radial-basis type kernel functions and called TPs. CMPs have been shown to allow statistical generalization based on given task parameters [7], but their learning has so far remained constrained to recording the torques of an execution with a stiff robot. Being a potentially dangerous method, it cannot be applied for autonomous database expansion.

The main contribution of this paper is fast and autonomous learning of CMPs. From a given desired kinematic behavior, the torques are learned iteratively through several repetitions until the error of compliant tracking is reduced below a predefined threshold. A good initial approximation reduces the number of needed iterations. This initial approximation is generated with statistical generalization from a database of learned motions. Such learning of joint torques for robots

Manuscript received September 21, 2017; revised April 9, 2018; accepted June 27, 2018. This paper was recommended for publication by Associate Editor D. Kulic and Editor A. Billard upon evaluation of the reviewers' comments. This work was supported in part by Sciex-NMSCH under Project 14.069 and in part by European FP7 ICT project Xperience under Grant 270273. *(Corresponding author: Tadej Petrič.)*

T. Petrič is with the Biorobotics Laboratory, École Polytechnique Fédérale de Lausanne, Lausanne CH-1015, Switzerland, and also with the Department for Automatics, Biocybernetics and Robotics, Jožef Stean Institute, Ljubljana 1000, Slovenia (e-mail: tadej.petric@ijs.si).

A. Gams and A. Ude are with the Department for Automatics, Biocybernetics and Robotics, Jožef Stean Institute, Ljubljana 1000, Slovenia (e-mail: andrej.gams@ijs.si; ales.ude@ijs.si).

L. Colasanto and A. J. Ijspeert are with the Biorobotics Laboratory, École Polytechnique Fédérale de Lausanne, Lausanne CH-1015, Switzerland (e-mail: luca.colasanto@hotmail.it; auke.ijspeert@epfl.ch).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2018.2861921

1552-3098 © 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

can be analogously observed in humans [9], where the kinematic trajectory is learned in Cartesian space (DMPs) and the task dynamics in the joint space (TPs).

This paper is a continuation of our work with substantial contributions relative to the previous conference paper [10]. The specific contributions of this paper, that go beyond our conference paper [10], are: the stability analysis; a more in-depth explanation of the method; an additional experiment, which depicts behavior during physical interaction; and extensive related work and discussion sections.

This paper is organized as follows. Section II describes the related work, separately detailing the topics of generalization, and learning of robot torque profiles. Section III describes the main contribution of this paper, i.e., autonomous learning of CMPs with an emphasis on learning robot joint torques. A stability analysis of this learning algorithm in Section III-C shows that the process is stable. Section IV describes the process of autonomously expanding a database of motions for statistical generalization. Results of experimental evaluation on a Kuka LWR-4 robot arm learning to perform a peg-in-hole task are presented in Section V. Section VI concludes this paper with a discussion.

II. Related Work

A. Generalization

Generalization of robot trajectories is a wide topic and can be considered from different aspects and domains of application. Many approaches deal with generalization of kinematic trajectories, but not so many with dynamics profiles. For this paper, the aspect of application in either kinematic or dynamic domain is specifically important. Another aspect of interest for this paper is the database generation and utilization. As the name implies, generalization utilizes a pre-existing database. On the one hand, this database can be generated by the user, but on the other hand, the robot could expand it on its own. The latter can be considered as an extrapolation–interpolation hybrid, where the database is extended autonomously beyond the reach of the original database, but then this extended database is used for generalization. Finally, the means of generalization depend on the trajectory representation.

The aforementioned DMPs, which constitute the kinematical part of the CMP, are one of the possible trajectory representations utilized in generalization of kinematic trajectories. DMPs already allow for a certain degree of generalization through the changing of the goal [8], which was effectively exploited in [11]. Yet changing the goal only explicitly changes the final position of the trajectory, while the evolution of the trajectory toward this position is left to the second-order attractor system of the DMPs. On the one hand, for generalization between a set of recorded trajectories, DMPs have been used as the means to represent the results of statistical generalization of both discrete (point-to-point) and periodic kinematic trajectories using locally weighted regression (LWR) [1]. On the other hand, generalization between the weights of the DMPs was implemented using Gaussian process regression (GPR) [12]. Both these approaches generate a new trajectory based on a task parameter, which is used as the query for the generalization. Similar task parameters were applied for learning of parameterized skills [13], where the authors propose adapting the DMP function approximator (the weights) of a single demonstration based on the task parameter. Contrary to using only one demonstration, many demonstrations were used to create a parametric attractor landscape in a set of differential equations by Matsubara *et al.* [14]. Parameterized skill memories [15] are also based on DMPs and enable task-specific retrieval of motion primitives, i.e., generalization from a low number of examples. Variations of DMPs have also been applied for generalization, for example, the mixture of motor primitives [16] was used obtain a task policy that is composed of several movement primitives weighted by their ability to generate successful movements in the given task context. It should be noted that the approach in [16] can autonomously update the weights.

DMPs have not been the only trajectory representation, or even the only dynamical systems applied for generalization. Because the approach in this paper is based on DMPs, we only briefly list possible alternatives. The approach of Calinon [17] is centered around task-specific Gaussian mixture models. The approach uses several frames of reference to describe the robot behavior in multiple coordinate systems, with the variations and correlations observed from the perspective of these different frames exploited to determine the impedance of the system with a linear quadratic regulator. GMMs were also applied in [18]. Hidden Markov models are another option [19]. See also [17] for an extensive list of papers on generalization.

While the above-mentioned approaches extensively explore generation of trajectories between existing database entries, database expansion has been researched less. Extrapolation is explicitly mentioned in [20], where the authors statistically encode movements in a task-parameterized mixture model, and derive an expectation-maximization algorithm to train it. In this paper, we explore how we can first create and then use these newly trained motions for an expanded database. The new motions can either be within the database or beyond it. The literature on such approaches is sparse [10], [20].

Generalization of dynamic variables, such as forces and torques, has also been studied previously. In an approach combining the kinematic DMP trajectories and dynamic coupling terms, [21] used generalization of the force-based coupling terms to generate new trajectories based on a given task parameter. The approach depends on user interaction to generate new database entries. Koropouli *et al.* [22] used the measured force-control policies with the view to executing constrained movements inside deformable and homogeneous environments. They considered policies where the output is force, but the input is the motion data. In essence, this is an inverse dynamics problem and consists of estimating the force responsible for a certain motion to be realized. Kober *et al.* [23] have shown how to acquire a task-level representation for motor primitives, along with the interaction forces, which gives it the ability to generalize to different situations. The generalization of both kinematic trajectories and torque profiles has been reported with the aforementioned CMPs in [7]. Our paper extends the approach in [7] with autonomous database extension and autonomous learning of torque profiles.

B. Torque Profile Learning

Accurate and compliant control of robots requires their dynamical models, but also the dynamical model of the task [24]. Specifically the dynamical model of the task can be hard to obtain [25]. Therefore, different methods, including biologically inspired methods, were proposed for robot control [26]. However, (relatively) novel robotic mechanisms, such as the Kuka LWR-4, are equipped with joint-torque sensors [27], which can be used to measure the torques during a task. This has led to the development of CMPs, first reported for periodic tasks [28]. CMPs are based on recording joint-torque profiles during a task, and using them as feed-forward torque components in the next repetition of the same task. It was shown in [7] that generalization of torque components of the CMP can be used to generate new torque profiles for a dynamically different task, or even for a task where the kinematic aspect has been altered. Furthermore, combinations of parts of several trajectories were shown to produce new trajectories in [29]. Thus, the complete dynamical model does not need to be derived, but learned task-specific torques can be used for these specific tasks.

Recording and feed-forwarding of torque trajectories was not only proposed in CMPs, but also in approaches that utilize the exact same possibility of measuring the joint torques. Calandra *et al.* [30] use tactile sensors to determine the force of contact with the environment on the iCub robot, and calculate the joint torques from the measured arm pose. These calculated joint torques are then used in a feed-forward manner for the control, just as is the case in CMPs. Similarly, Steinmetz *et al.*[31] recorded joint torques along the kinematic trajectory, encoded as a DMP, and used these torques as a feed-forward signal for the controller to increase the accuracy in the next execution of the in-contact task.

The need to first record the CMP torque profile, which comes from an exact—and therefore stiff robot execution of the task using high gains [32], has made the application of CMPs limited. Note that stiff executions might not be safe for the robot and/or humans/objects in interaction [33]. Stiff execution also means that any extension of the CMP database for generalization requires an operator. In this paper, we propose an approach of autonomously learning the torque profile of the CMPs. Initial partial results were presented in [10].

Other approaches have been proposed for torque learning. For example, Nguyen-Tuong and Peters [34], [35] used local GPR for online dynamic model learning. Thus, they improved the accuracy of the model and avoided high feedback gains. Their approach requires the availability of a large quantity of data in order to learn a complete dynamic model, and not only task-specific torques. Learning of torques for a specific task can be utilized using ILC [5]. For example, Schwarz and Behnke [36] used ILC to learn motor and friction models. Kronander *et al.* [37] used incremental motion learning to locally update dynamical systems. Gautier *et al.* [38] proposed an iterative learning identification and control method for dynamic robot control. Levine *et al.* [39] have optimized linear-Gaussian controllers under unknown dynamics by iteratively fitting local linear dynamics models, with a background dynamics distribution acting as a prior to reduce the sample complexity. Thus, they could outperform model-free approaches and obtain local, time-varying models, which allowed them to perform dynamic tasks where the model is difficult to learn. An example of such is the peg-in-hole task, which serves as the use-case in our paper too.

III. Compliant Movement Primitives

CMPs $h(t)$ are defined as a combination of kinematic trajectories encoded in DMPs and corresponding task-specific dynamics encoded in TPs

$$\boldsymbol{h}(t) = [\ddot{\boldsymbol{p}}_d(t), \dot{\boldsymbol{p}}_d(t), \boldsymbol{p}_d(t),\ \boldsymbol{\tau}_f(t)]. \tag{1}$$

where $\ddot{\boldsymbol{p}}_d(t), \dot{\boldsymbol{p}}_d(t), \boldsymbol{p}_d(t)$, are the desired task-space acceleration, velocity, and position trajectories, respectively, encoded in the DMPs. The $\boldsymbol{\tau}_f$ are the corresponding task-specific joint torques encoded in TPs. In the proposed approach, a two-stage process is used to obtain the CMPs. First, the kinematic motion trajectories are obtained either by human demonstration, or are predefined, and encoded as DMPs [8], [40]. Then, the corresponding torques are obtained using recursive regression based on error learning and encoded as TPs, similar as in [10].

A. Motion Trajectories

A short recap of the DMPs [8] for encoding the motion is given first. For details on Cartesian DMPs including rotations refer to [41]. Within this framework the trajectories can be given in either joint or task space, and every degree of freedom is described by its own dynamic system. The movement trajectory for each DOF is described by the following system of nonlinear differential equations:

$$v\dot{z} = \alpha_z\left(\beta_z\left(g - y\right) - z\right) + f(s) \tag{2}$$

$$v\dot{y} = z. \tag{3}$$

Here, y is the trajectory, z is temporally scaled velocity, v is the time constant, and α_z and β_z are positive constants that ensure critical damping so that the system monotonically converges toward goal g, and the nonlinear part $f(s)$ is defined as a linear combination of radial basis function $\Psi_i(s)$, defined as

$$f(s) = \frac{\sum_{i=1}^{N} w_i \Psi_i(s)}{\sum_{i=1}^{N} \Psi_i(s)} s, \tag{4}$$

$$\Psi_i(s) = \exp(-d(s - c_i)^2) \tag{5}$$

where N is the number of kernel functions (kernels) weighted with $\mathbf{w}$, which defines the actual shape of the encoded trajectory. The variables d and c_i define the widths and centers of the kernels, respectively. Note that $f(s)$ is not directly time dependent. Therefore, a phase variable s with initial value $s(0) = 1$ is used

$$v\dot{s} = -\alpha_s s \tag{6}$$

where α_s is the phase parameter. If not stated otherwise, we used $\alpha_s = 2$.

To encode the DMPs, the weight vector $\mathbf{w}$ needs to be learned, for example, with incremental LWR. The target data for fitting is

$$f_t = v^2\ddot{p}_d + \alpha_z v\dot{p}_d - \alpha_z\beta_z(g - p_d). \tag{7}$$

Given the target f_t, w_i is updated for each time step j as

$$w_{i,j+1} = w_{i,j} + \Psi_i P_{i,j+1} e_j \tag{8}$$

$$P_{i,j+1} = \frac{1}{\lambda}\left(P_{i,j} - \frac{P_{i,j}^2}{\frac{\lambda}{\Psi_i} + P_{i,j}}\right) \tag{9}$$

$$e_j = f_{t,j} - w_{i,j}. \tag{10}$$

Here P_i is the inverse covariance. The regression starts with $w_i = 0$ and $P_i = 1$. λ is the forgetting factor set to $\lambda < 1$, typically at $\lambda = 0.99$. Alternatively, batch regression can be used for more accurate reproduction when training data with low noise are available [1].

B. Joint Torque Trajectories

The TPs are learned recursively while executing the encoded DMP motion with low-gain impedance control. The desired motion $\ddot{\boldsymbol{p}}_d$, $\dot{\boldsymbol{p}}_d$, $\boldsymbol{p}_d$ encoded in DMPs is executed using the following control law:

$$\boldsymbol{\tau}_u = \mathbf{J}^T\left(\mathbf{K}_p\boldsymbol{e}(s) + \mathbf{K}_d\dot{\boldsymbol{e}}(s) + \mathbf{K}_i\ddot{\boldsymbol{e}}(s)\right) + \mathbf{N}\mathbf{K}_n\dot{\boldsymbol{q}}_n + \boldsymbol{\tau}_f(s) \tag{11}$$

where $\mathbf{J}^T$ is the Jacobian transpose, $\mathbf{N}$ is the dynamically consistent null space as in [42], $\dot{\boldsymbol{q}}_n$ is the joint velocity vector, $\boldsymbol{e}$, $\dot{\boldsymbol{e}}$ and $\ddot{\boldsymbol{e}}$ are the differences between desired and actual position $\boldsymbol{p}$, velocity $\dot{\boldsymbol{p}}$ and acceleration $\ddot{\boldsymbol{p}}$, respectively, and $\mathbf{K}_p$, $\mathbf{K}_d$, $\mathbf{K}_i$, and $\mathbf{K}_n$ are the constant gain matrices selected such that the robot behaves compliantly, i.e., set to match the low-impedance control requirements. The $\boldsymbol{\tau}_f(s)$ is vector of feed-forward torque trajectories $\boldsymbol{\tau}_f(s) = [\tau_{f,1}(s), \tau_{f,2}(s), \ldots, \tau_{f,j}(s), ..., \tau_{f,M}(s)]^T$, where M is the number of DOF. For one DOF, $\tau_{f,j}(s)$ is given by

$$\tau_{f,j}(s) = \begin{cases} \dfrac{\sum_{i=1}^{N}\psi_i(s)w_{i,j}^{\tau}}{\sum_{i=1}^{N}\psi_i(s)} & s \geq s_\epsilon \\ \tau_{f,j}(s_\epsilon) & s < s_\epsilon \end{cases} \tag{12}$$

where the phase s is common with the DMPs and s_ϵ denotes the final value of the phase variable for the encoded learned torque signal. This ensures that the final torque value is maintained, even if CMPs are executed beyond the final learned point. Note that for each movement variation, including movement speed, a new TP has to be obtained. A library of motion can be built by storing TPs. Furthermore, statistical generalization or graph search can be used to generate new TPs, which were previously not explicitly learned. For details see [7] and [29].

However, in cases when the desired movement are not covered by the database, the approaches from [7] and [29] will not work. To expand the database, we previously had to learn a new TP with a stiff robot, i.e., with high-gain impedance control, which required operator input. In this paper, we propose a new method that can autonomously learn new CMPs. The proposed method recursively updates the weights $w^\tau_{i,j}$ of TPs. Similarly as for DMPs, the recursive regression method is given by

$$w^\tau_{i,j}(t+1) = w^\tau_{i,j}(t) + \psi_i P_{i,j}(t+1)\epsilon_j(t) \tag{13}$$

$$P_{i,j}(t+1) = \frac{1}{\lambda}\left(P_{i,j}(t) - \frac{P^2_{i,j}(t)}{\frac{\lambda}{\psi_i} + P_{i,j}(t)}\right) \tag{14}$$

where $P_{i,j}$ is the covariance. The initial parameters are set to $P_i = 1$, $w_i = 0$, and $\lambda = 0.995$. The TP update criterion is defined similarly as in feedback-error learning [43], where the joint torques are learned based on proprioceptive errors in Cartesian space. By using the following definition, the error of motion is defined as:

$$\boldsymbol{\epsilon}(t) = \mathbf{J}^T\left(\alpha_t(\boldsymbol{p}_d(t) - \boldsymbol{p}(t)) + \beta_t(\dot{\boldsymbol{p}}_d(t) - \dot{\boldsymbol{p}}(t))\right). \tag{15}$$

Parameters α_t and β_t define the update rate, and $(.)_d$ denotes the desired values. Note that the error vector is defined as $\boldsymbol{\epsilon} = [\epsilon_1, \epsilon_2, \ldots, \epsilon_j, ..., \epsilon_M]$, where M is the number of DOFs. Each DOF is updated separately with (13) and (14). If feedback error is defined as in (15), the TPs will converge toward joint torques that match the motion behavior encoded in DMPs. Note that other formulations are possible, for example, additional constrains can be added.

C. Stability Analysis

Stability analysis of the proposed approach is based on [44]. Notations Y_d, Y_i, U_i, and E_i denote the reference signals, output signals, control signals, and error signals, respectively, at the ith iteration. P and C denote the transfer function of the linearized robot dynamics and the transfer function of the feedback controller, respectively. Q denotes the transfer function of recursive regression and K denotes the gain. Note that K is also influenced by the width h and number N of Gaussian kernel functions in (12). By assuming that the transfer function for regression is known and that the feedback system without the iteration loop is already stable, then the control signal update law is given by

$$U_i = U'_i + CE_i \tag{16}$$

where U'_i is the current regression update based on a previous state, given by

$$U'_i = U'_{i-1} + KQE_i. \tag{17}$$

It is called current regression update because the current cycle tracking error E_i is involved in learning. For the given control task Y_d, the convergence condition are as follows:

$$\begin{aligned}
E_i &= Y_d - Y_i \\
&= Y_d - PU_i \\
&= Y_d - P(U'_i + CE_i) \\
&= Y_d - P(U'_{i-1} + KQE_i + CE_i) \\
&= Y_d - P(U'_{i-1} + CE_{i-1} - CE_{i-1} + KQE_i + CE_i) \\
&= Y_d - P(U_{i-1} - CE_{i-1} + KQE_i + CE_i) \\
&= Y_d - Y_{i-1} - P(-CE_{i-1} + KQE_i + CE_i) \\
&= E_{i-1} - P(-CE_{i-1} + KQE_i + CE_i)
\end{aligned}$$

$$\Rightarrow \frac{E_i}{E_{i-1}} = \frac{1 + PC}{1 + PC + PKQ} \tag{18}$$

$$\Rightarrow \| \frac{E_i}{E_{i-1}} \| \models \| \frac{1 + PC}{1 + PC + PKQ} \| \leq \gamma < 1 \tag{19}$$

Here, the norm $\| \circ \|$ is the infinity norm for all frequencies $\omega \in \Omega$, where Ω denotes the frequency band of interest. Stability conditions imply that tracking error E_i tends to 0 for $i \to \infty$. The design of the controller can be thus performed in subsequent steps.

1) Check if stability condition is fulfilled by, e.g., Bode or Nyquist plot.
2) Tune the parameters of the transfer function C and K.

The stability proof of the proposed approach is demonstrated on a one-degree-of-freedom mechanism where we assume that the feedback system without the iteration loop is stable. Here, we consider a third-order system transfer function P for the mechanism with the PD controller for C. Since it is difficult to derive transfer function Q of the recursive regression given by (13) and (14), we used identification instead. The results of the identification showed that for a given set of parameters, i.e., the number of kernel function $N = 25$, their width $d = 1$ and the forgetting factor $\lambda = 0.995$, the recursive regression can be approximated as follows:

$$Q = \frac{12.44s^2 + 248.9s + 2738}{s^3 + 8.93s^2 + 180.4s + 338.7}. \tag{20}$$

The transfer function was estimated with the accuracy of 98.8%, final prediction error of 0.001 and mean square error of 0.001. The transfer function for K is derived from (15), using gains $\alpha_t = 20$ and $\beta_t = 20$. The Bode analysis of the system given with (19) is shown in Fig. 1, where we can see that the stability condition of (19) is fulfilled. Stability condition further implies that the tracking error E_i tends to 0 for $i \to \infty$.

If not stated otherwise, the same parameter set was used in the rest of the paper, i.e., $\alpha_t = 20$, $\beta_t = 20$, $N = 25$, $d = 1$, and $\lambda = 0.995$.

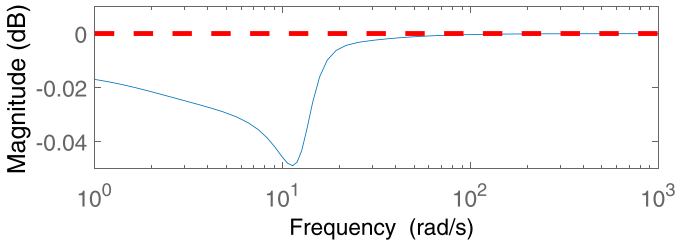

Fig. 1. Bode analysis of the transfer function (19), showing that the stability criteria of the proposed algorithem is fulfilled.

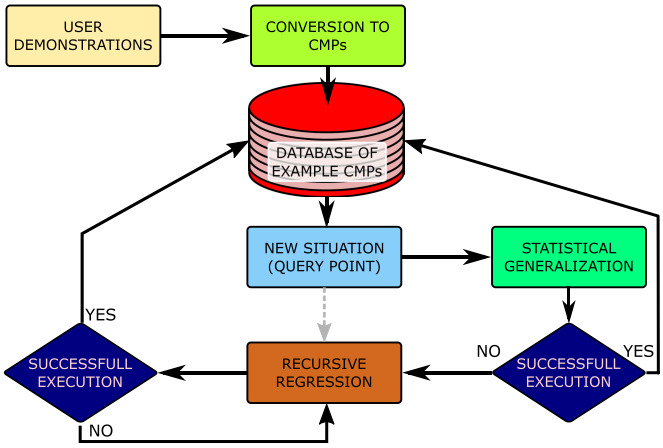

Fig. 2. Learning of CMPs using bootstrapping in the motor space.

IV. Autonomous Database Expansion

Iterative learning of CMPs simplifies the execution of dynamically versatile tasks, while ensuring at the same time accurate and compliant execution of the motion. However, since torques are usually not linearly scalable, TPs have to be learned for every variation of the task, which includes, for example, different payloads, goals, and even speeds. To avoid new learning for each task variation, or to at least significantly accelerate it, we propose using statistical generalization, which can generate good approximations of the TPs based on a given query point if sufficient training data are available. In case when executed generalized TPs satisfy the given error criterion, for example, the trajectory-tracking accuracy, they can be immediately added to the database of motion. If not, i.e., if the tracking accuracy is not sufficient, the recursive regression method (13)–(15) can be applied. Note that in such cases the generalized TPs are used for the initial approximation. This vastly reduces the number of needed iterations for learning. Fig. 2 shows the basic schema of the approach.

The task-space error, which is used in (15) to ensure that a robot accurately tracks the desired task-space trajectory encoded by a DMP, determines if a new TP should be added to the database

$$e_p = \sum_{j=1}^{I} ||\boldsymbol{e}(j)||. \tag{21}$$

Here, $\boldsymbol{e}(j)$ is the vector of absolute difference between the actual task-space position $\boldsymbol{p}$, and the desired position encoded in the DMP $\boldsymbol{p}_d$. I is the number of steps for one movement execution (iteration).

The database $\boldsymbol{H}_x^{\mathrm{TP}}$ for a given set of L examples is given by

$$\boldsymbol{H}_x^{\mathrm{TP}} = \{\boldsymbol{w}_{\tau\,k}, \boldsymbol{c}_k\},\ k = 1, 2, \ldots, L \tag{22}$$

where a TP, defined by weights $\boldsymbol{w}_\tau$ is used together with associated DMPs to execute a task, defined by the query point $\boldsymbol{c}$. Note that the task is executed in a compliant manner using low-feedback gains. By using GPR for statistical regression as in [7]

$$\boldsymbol{F}_{\boldsymbol{H}_x^{\mathrm{TP}}} : \boldsymbol{c} \longmapsto [\boldsymbol{w}_q, \boldsymbol{w}_\tau] \tag{23}$$

we can compute the initial TP parameters for the given query $\boldsymbol{c}$, i.e., for the task variation. Note that generalization can also be used with DMPs (see [1]).

The learning process for a new TP is repeated as long as $e_p > e_c$, where e_c is the given threshold. Once this criterion is met, the new TP is added to the database of motion $\boldsymbol{H}_x^{\mathrm{TP}}$. With the proposed approach, the database of CMPs can be autonomously expanded.

Fig. 3. Experimental setup for the peg-in-hole task performed with Kuka LWR-4 robot.

Note that this kind of database expansion assumes that the kinematic aspect of the task is either known or can be computed, for example, with statistical generalization as in [1]. See also the discussion section.

V. Experimental Evaluation

The proposed method was evaluated on the Kuka LWR-4 robot, where we demonstrate the peg-in-hole task, which includes interaction with the environment. Here, we first show the ability to learn CMPs, followed by the evaluation of the accuracy of generalized TPs using a leave-one-out cross-validation approach.

If CMPs are combined with low-gain impedance control, the interaction forces in case of collision with unforeseen objects will be significantly smaller compared to high-gain impedance control with similar tracking performance. Note that to achieve similar tracking performance with only impedance control, the robot should be practically stiff. For in-depth tracking and compliance analysis of CMPs please see [7].

To investigate the performance in case of collisions, we chose the peg-in-hole task. Here, the robot needs to learn the policy that compensates for the tool dynamics, i.e., the dynamics of the peg insertion, and then interact with the environment. The peg insertion also requires that a certain degree of force is applied for successful completion of the task, which further implies that the robot needs to learn new task dynamics that includes friction for each location of the hole.

The experimental setup for the peg-in-hole task is shown in Fig. 3. The task of the robot was to move from an initial position and insert the peg. Here, we assume that the desired motion encoded by a DMP is accurate enough for peg insertion. This means that if the robot could accurately track the DMP-provided trajectory, the peg insertion would be successful. However, due to the limitation of high-gain feedback control, i.e., high forces when in contact, this type of control cannot be used for pegs and holes with tight tolerances. However, in our proposed approach, the robot can gradually learn the corresponding torques for performing the task while being compliant.

To demonstrate the ability of learning and gradually building a database of motion, we selected 13 different positions for peg insertion,

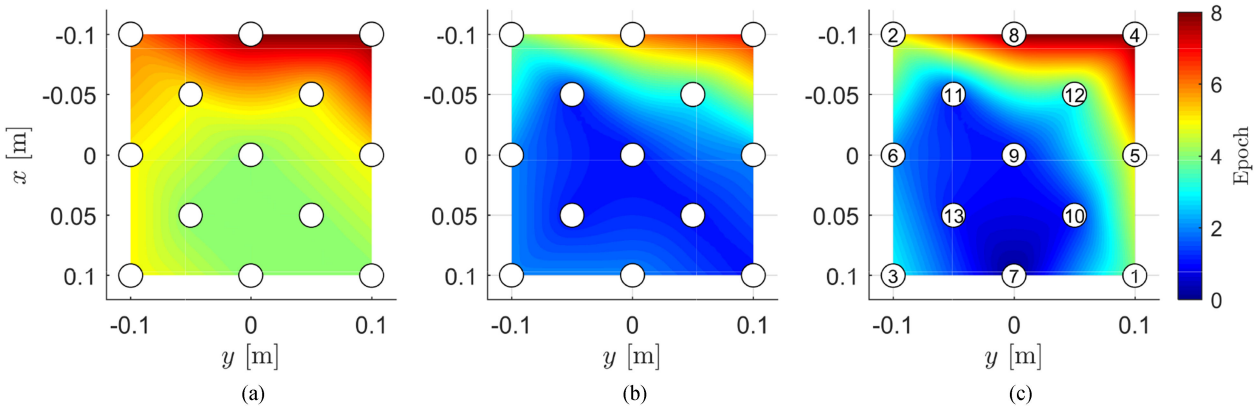

Fig. 4. Comparison of TP learning for the peg-in-hole task. (a) Without a database—five learning epochs on average. (b) Leave-one-out cross validation—two epochs on average. (c) Gradually building the database—two to three epochs on average. The numbers in circles denote the sequence of learning, and therefore also the sequence of building the database.

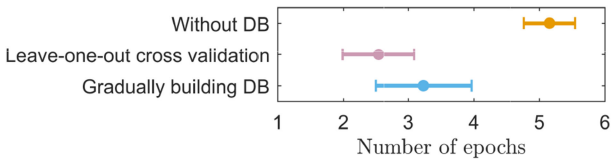

Fig. 5. Mean value and standard error of required epochs for TP learning: without a database, with the database (leave-one-out cross validation), and when gradually building the database.

all on the same plane. The locations of the holes in robot coordinate system can be seen in Fig. 4. To learn the corresponding TPs we again used an algorithm explained in Section III-B.

Three different cases were implemented for comparison. First, learning of each possible position of the hole without using generalization for the initial TP approximation. We denoted this approach by (a) in Fig. 4, where we can notice that on the average at least five learning epochs were needed to properly learn the skill.

Then, we applied the leave-one-out cross validation, where the selected position for the peg insertion was excluded from the database. All other movements were used for generalization of initial TP approximation. This approach was denoted by (b) in Fig. 4. We can see in the middle plot that on average two epochs were needed to properly obtain the required skill for peg insertion.

Finally, we show the performance of the proposed approach while gradually building the database of motion. Note that in this experiment the database was empty at the beginning. This approach was denoted by (c) in Fig. 4. This sequence was chosen because it shows that at the beginning, for cases from 1 to 4, the proposed approach is able to successfully expand the database autonomously. In the following cases, from 5 to 14, it shows that learning improves significantly compared to learning without using prior knowledge. We can see by comparing (b) and (c) that for cases 5 and onward, the number of epochs needed to successfully acquire the skill is similar. Note that for case 5 only movements from 1 to 4 were included in the database.

Besides the initial TP state, in general, the learning gains α_t and β_t [see (15)] also influence the learning rate. Setting these gains too high might potentially destabilize the system or if they are set too low, the rate of learning might not be effective. With α_t and β_t set to predefined constants, the number of required epochs for TP learning depends only on its initial estimate. The effect of using prior knowledge from a database for initial TP estimation is depicted in Fig. 5. Here, we can see that the learning is almost twice as fast if the initial TP is approximated using statistical generalization.

VI. Discussion

Autonomous generation of trajectories can improve the ability of robots to act without constraints in unstructured environments. In this sense, the expansion of a trajectory database for generalization is a step in the right direction. On the one hand, we have shown that we can generate trajectories between the given data, i.e., make the database more dense, but on the other hand, we have also shown that we can expand the database beyond its borders, as in extrapolation.

Expansion of a trajectory database in this paper cannot be viewed as extrapolation *per se*, since critically, the kinematic aspect of motion, i.e., the DMP kinematic trajectory, has to be present. When making the database more dense, this trajectory can be the result of generalization, which will preserve the properties of motion [1]. On the one hand, for the database expansion, it can be as easy as demanding that the motion is straight, which sufficiently defines the properties of motion. On the other hand, complex motions that *resemble* the movements in the database cannot be obtained so easily. This is somewhat analogous to human motor learning, where it seems to be far easier to optimize an aspect of a previously acquired motion or even a paradigm, than it is to come up with something completely new.

In our case, we applied iterative learning to learn the dynamical aspects, i.e., the TP of the desired motion, which provides a new input for the database. In contrast to explorative learning, e.g., reinforcement learning, the reward for learning is the tracking error, which is signed (plus or minus). Reinforcement learning, however, is a method that could supplement iterative learning, as it can explore with far less constraints. Yet, explorative methods typically take considerably more iterations, one or two orders of magnitude more, to achieve the same results as iterative learning. Another aspect of consideration is that while using such methods, the results will optimize the reward, but the motion might not resemble the other motions in the database. Such results would not be useful for subsequent generalization. Additional constraints could direct the learning, but designing of such rewards can be extremely complex. It is therefore preferable to design the kinematic motion and apply iterative learning when applicable.

It should be stressed that the learning of dynamical aspects of motion, i.e., the TPs, is not trivial, as one has to take into consideration safety aspects as well as the nonlinearity of the search space. Hence, autonomous learning of TPs is another important contribution of this paper. In combination with generalization, which provides initial estimates, autonomous learning becomes very fast and provides the means to genuinely autonomously expand initial trajectory databases.

VII. Conclusion

The proposed methods were shown to be applicable in contact with the environment. They extend the level of applicability of the CMP paradigm [7] to real-world scenarios. The results have shown that the database expansion can be autonomous, fast and applicable with different constraints, such as the afore mentioned contact with the environment. In the future, we will investigate if explicit dynamics models could be used as prior knowledge for the coarse estimation of the initial TPs and we will explore more complex constraints, such as stability aspects on humanoid robots with a free-floating base.

References

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

[2] J. Kober, A. Wilhelm, E. Oztop, and J. Peters, "Reinforcement learning to adjust parametrized motor primitives to new situations," *Auton. Robots*, vol. 33, pp. 361–379, 2012.

[3] M. Kalakrishnan, L. Righetti, P. Pastor, and S. Schaal, "Learning force control policies for compliant manipulation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2011, pp. 4639–4644.

[4] M. Hazara and V. Kyrki, "Reinforcement learning for improving imitated in-contact skills," in *Proc. IEEE-RAS 16th Int. Conf. Humanoid Robots*, Nov. 2016, pp. 194–201.

[5] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *IEEE Control Syst.*, vol. 26, no. 3, pp. 96–114, Jun. 2006.

[6] A. Albu-Schaffer *et al.*, "Soft robotics," *IEEE Robot.Automat. Mag.*, vol. 15, no. 3, pp. 20–30, Sep. 2008.

[7] M. Deniša, A. Gams, A. Ude, and T. Petrič, "Learning compliant movement primitives through demonstration and statistical generalization," *IEEE/ASME Trans. Mechatronics*, vol. 21, no. 5, pp. 2581–2594, Oct. 2016.

[8] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Comput.*, vol. 25, pp. 328–73, 2013.

[9] J. W. Krakauer, M. F. Ghilardi, and C. Ghez, "Independent learning of internal models for kinematic and dynamic control of reaching," *Nature Neurosci.*, vol. 2, no. 11, pp. 1026–31, Nov. 1999.

[10] T. Petrič, L. Colasanto, A. Gams, A. Ude, and A. Ijspeert, "Bio-inspired learning and database expansion of compliant movement primitives," in *Proc. IEEE-RAS 15th Int. Conf. Humanoid Robots*, Nov. 2015, pp. 346–351.

[11] R. Lioutikov, O. Kroemer, J. Peters, and G. Maeda, "Learning manipulation by sequencing motor primitives with a two-armed robot," in *Proc. 13th Int. Conf. Intell. Auton. Syst.*, 2014, pp. 1601–1611.

[12] D. Forte, A. Gams, J. Morimoto, and A. Ude, "On-line motion synthesis and adaptation using a trajectory database," *Robot. Auton. Syst.*, vol. 60, no. 10, pp. 1327–1339, 2012.

[13] F. Stulp, G. Raiola, A. Hoarau, S. Ivaldi, and O. Sigaud, "Learning compact parameterized skills with a single regression," in *Proc. 13th IEEE-RAS Int. Conf. Humanoid Robots*, Oct. 2013, pp. 417–422.

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

[15] R. Reinhart and J. Steil, "Efficient policy search with a parameterized skill memory," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2014, pp. 1400–1407.

[16] K. Muelling, J. Kober, O. Kroemer, and J. Peters, "Learning to select and generalize striking movements in robot table tennis," *Int. J. Robot. Res.*, vol. 32, no. 3, pp. 263–279, 2013.

[17] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," *Intell. Serv. Robot.*, vol. 9, no. 1, pp. 1–29, 2015.

[18] S. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with gaussian mixture models," *IEEE Trans. Robot.*, vol. 27, no. 5, pp. 943–957, Oct. 2011.

[19] D. Lee and C. Ott, "Incremental kinesthetic teaching of motion primitives using the motion refinement tube," *Auton. Robots*, vol. 31, no. 2, pp. 115–131, 2011.

[20] S. Calinon, T. Alizadeh, and D. Caldwell, "On improving the extrapolation capability of task-parameterized movement models," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 610–616.

[21] A. Gams, M. Deniša, and A. Ude, "Learning of parametric coupling terms for robot-environment interaction," in *Proc. IEEE-RAS 15th Int. Conf. Humanoid Robots*, Nov. 2015, pp. 304–309.

[22] V. Koropouli, S. Hirche, and D. Lee, "Generalization of force control policies from demonstrations for constrained robotic motion tasks," *J. Intell. Robot. Syst.*, vol. 80, no. 1, pp. 133–148, 2015.

[23] J. Kober, M. Gienger, and J. J. Steil, "Learning movement primitives for force interaction tasks," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2015, pp. 3192–3199.

[24] J. Peters, K. Mülling, D. Kober, Jensand Nguyen-Tuong, and O. Krömer, "Towards motor skill learning for robotics" in *Robotics Research: The 14th International Symposium (ISRR)*. New York, NY, USA: Springer-Verlag, 2011, pp. 469–482.

[25] T. Petrič, B. Curk, P. Cafuta, and L. Žlajpah, "Modelling of the robotic Powerball: A nonholonomic, underactuated and variable structure-type system," *Math. Comput. Model. Dyn. Syst.*, vol. 16, no. 4, pp. 327–346, Nov. 2010.

[26] D. W. Franklin and D. M. Wolpert, "Computational mechanisms of sensorimotor control," *Neuron*, vol. 72, no. 3, pp. 425–442, 2011.

[27] R. Bischoff *et al.*, "The KUKA-DLR Lightweight Robot arm - a new reference platform for robotics research and manufacturing," *41st Int. Symp. Robot., 2010 6th Ger. Conf. Robot.*, 2010, pp. 1–8.

[28] T. Petrič, A. Gams, L. Žlajpah, and A. Ude, "Online learning of task-specific dynamics of periodic tasks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Chicago, IL, USA, 2014, pp. 1790–1795.

[29] M. Denisa, T. Petrič, T. Asfour, and A. Ude, "Synthesizing compliant reaching movements by searching a database of example trajectories," *Int. Conf. Humanoid Robots*, 2013, pp. 540–546.

[30] R. Calandra, S. Ivaldi, M. Deisenroth, and J. Peters, "Learning torque control in presence of contacts using tactile sensing from robot skin," in *Proc. IEEE-RAS 15th Int. Conf. Humanoid Robots*, Nov. 2015, pp. 690–695.

[31] F. Steinmetz, A. Montebelli, and V. Kyrki, "Simultaneous kinesthetic teaching of positional and force requirements for sequential in-contact tasks," in *Proc. IEEE-RAS 15th Int. Conf. Humanoid Robots*, Nov. 2015, pp. 202–209.

[32] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*, vol. 3. Hoboken, NJ, USA: Wiley, 2006.

[33] K. Kronander and A. Billard, "Learning compliant manipulation through kinesthetic and tactile human-robot interaction," vol. 7, pp. 367–380, 2014.

[34] D. Nguyen-Tuong and J. Peters, "Learning robot dynamics for computed torque control using local gaussian processes regression," in *Proc. ECSIS Symp. Learn. Adapt. Behav. Robot. Syst.*, Edinburgh, U.K., 2008, pp. 59–64.

[35] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: A survey," *Cogn. Process.*, vol. 12, no. 4, pp. 319–40, Nov. 2011.

[36] M. Schwarz and S. Behnke, "Compliant robot behavior using servo actuator models identified by iterative learning control," in *Proc. 17th RoboCup Int. Symp.*, Eindhoven, The Netherlands, 2013, pp. 207–218.

[37] K. Kronander, M. Khansari, and A. Billard, "Incremental motion learning with locally modulated dynamical systems," *Robot. Auton. Syst.*, vol. 70, pp. 52–62, 2015.

[38] M. Gautier, A. Jubien, and A. Janot, "Iterative learning identification and computed torque control of robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Nov. 2013, pp. 3419–3424.

[39] S. Levine and P. Abbeel, "Learning neural network policies with guided policy search under unknown dynamics," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. New York, NY, USA: Curran Associates, Inc., 2014, pp. 1071–1079.

[40] S. Schaal, P. Mohajerian, and A. Ijspeert, "Dynamics systems vs. optimal control–a unifying view," *Prog. Brain Res.*, vol. 165, no. 1, pp. 425–45, Jan. 2007.

[41] A. Ude, B. Nemec, T. Petric, and J. Morimoto, "Orientation in cartesian space dynamic movement primitives," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2014, pp. 2997–3004.

[42] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE J. Robot. Automat.*, vol. 3, no. 1, pp. 43–53, Feb. 1987.

[43] M. Kawato, "Feedback-errror-learning neural network for supervised motor learning," in *Proc. Adv. Neural Comput.*, R. Eckmiller, Ed. New York, NY, USA: Elsevier, 1990, pp. 365–372.

[44] J.-X. Xu, S. K. Panda, and T. H. Lee, *Real-Time Iterative Learning Control(Advances in Industrial Control)*. New York, NY, USA: Springer-Verlag, 2009.