Transfer of Contact Skills to New Environmental Conditions

Aljaž Kramberger[1], Andrej Gams[1], Bojan Nemec[1],
Casper Schou[2], Dimitrios Chrysostomou[2], Ole Madsen[2], Aleš Ude[1]

Abstract— **Robots operating in contact with the environment should typically take into account the knowledge of both position/orientation trajectories as well as the accompanying force/torque profiles for successful execution. Pure position control is not appropriate because even small errors in the desired trajectory can cause significant forces at the contact points. In this paper we present a method that computes an appropriate control policy for a given condition of a contact task, with the peg-in-hole (PiH) assembly tasks as example use-cases. Our method is based on statistical generalization of successfully recorded executions at different values of the external condition. The major novelty of the method is that it provides not only generalized position and orientation trajectories, but a complete skill, consisting of desired position/orientation trajectories and the accompanying force/torque profiles. To improve the execution of the skill after generalization, we combine the proposed approach with an adaptation method to refine the newly generated movement. The versatility of the proposed approach was shown by applying it to firstly, two different types of robot arms: a humanoid 7-axis Kuka LWR-4 arm and a 6-axis industrial Universal robot UR5 arm and secondly, two different peg-in-hole problems: insertion of a square peg and insertion of a round peg.**

I. INTRODUCTION

Nowadays, many small and medium enterprises are trying to shift their production lines from mass production to mass customization of products. The impact of this trend is that the production systems have to adapt to handle more product variations, shorter life cycles, and smaller batch sizes. One of the main enablers of this transition is robotics and specifically robotic systems that can cope with uncertainty in interactions with humans, handle a variety of different tasks, and are able to be reprogrammed fast by non-robot experts when a new task in the factory arises.

The backbone of our approach to teaching robot skills is Programming by Demonstration (PbD) and statistical learning of robot trajectories. Traditionally, programming by demonstration has been applied to robots such as humanoids because the transfer of human body motion to humanoids is more natural than the tranfer to other types of robots. However, recent work on learning by demonstration [1], [2], e. g., through kinesthetic guiding [3]–[5] of robotic arms, has provided the necessary framework for knowledge transfer not only to humanoids but also to more standard industrial robots because with kinesthetic guiding we can avoid the problem of motion transformation. This is the case also with the method proposed in this paper, which is suitable for learning of robot operations in contact with the environment and has been tested both on an anthropomorphic arm as well as on a conventional 6-axis industrial arm.

More specifically, we developed a new approach for generalization of contact-rich robotic skills and applied it to a classical robotic assembly task: the peg-in-hole (PiH) insertion, although other applications are also possible. The gist of our approach is in adapting to an external condition, which defines the task and can result in significant changes in the required robot motion, which cannot be handled by feedback control and fine adaptation only. In the PiH case, this external condition – used as a query into a library of demonstrated movements – can for example be the depth of the hole.

To successfully execute the PiH task, a robot must perform trajectories which generate force/torque profiles that are suitable for a successful PiH execution. Thus the robot cannot execute such tasks using position control only because even small errors in the desired trajectory can cause significant deviations from the desired forces and torques [6]. However, when the external condition of the task, e. g. the depth of the hole, changes, new position/orientation trajectories as well as new force/torque profiles are needed. The major novelty of the proposed method is that it allows the transfer of the learned skill that involves position and force control from multiple known conditions of the task, to new, not previously known conditions, therefore providing all the required trajectories and force/torque profiles. Previously developed adaptation methods can then be applied to refine the generalized motion [6]–[8].

In a typical learning by demonstration setup, position and orientation trajectories are recorded. However, for tasks that involve contacts with the environment, e. g. PiH, forces and torques also have to be taken into account.

A widely accepted approach in PbD is to encode the trajectories with dynamic movement primitives (DMP) [9]. This framework enables efficient modulation of trajectories while they are being executed, both spatially and temporally, because they are not explicitly time dependent. Schaal et al. [10] explain that direct time dependence is often inappropriate, since it does not allow to change the speed of execution. Furthermore, it does not allow stopping or restarting robotic movements in the event of unforeseen disturbances during

The research leading to these results has been funded by the European Community's Seventh Framework Programme FP7/2007-2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 600578, ACAT and EU Horizon 2020 Programme grant no. 680431, ReconCell.

[1]Humanoid and Cognitive Robotics Lab, Dept. of Automatics, Biocybernetics and Robotics, Jožef Štefan Institute, Ljubljana, Slovenia. aljaz.kramberger@ijs.si

[2]Robotics & Automation Group, Dept. of Mechanical and Manufacturing Engineering, Aalborg University, Aalborg, Denmark. dimi@m-tech.aau.dk

Published in the proceedings of IEEE-RAS International Conference on Humanoid Robots (Humanoids), Cancun, Mexico, pp. 668-675, 2016.

execution of the desired task. However, the ability to adapt speed is important, because speed changes during the PiH execution are often required in order to raise the success of the task.

Our approach requires both positions/orientations and force/torque profiles. Similar approaches have been proposed before. Pastor et al. [11] introduced a method for real-time adaptation of learned trajectories depending on measured sensory data, where a force controller modified the accelerations of the DMP to comply with the previously measured force profile. On the other hand, we proposed methods where arbitrary desired force profiles were observed through a combination with iterative learning control [6], [12]. Koropouli et al. [13] have advanced beyond adaptation and proposed policy generalization approaches employed for motion-based force control policies. They learn the policy and policy difference data by locally weighted regression (LWR) and combine them in a superposition fashion to estimate the policy at new inputs. In our approach, we use statistical generalization to generate complete movement and force-torque trajectory estimates for new queries (inputs).

Statistical generalization uses accumulated data in order to generate new knowledge, through a statistical process. Several methods were successfully applied in robotics, for example, Locally Weighted Regression (LWR) [14], Locally Weighted Projection Regression (LWPR) [15], and Gaussian Process Regression (GPR) [16]. Generalization was also applied to different motion representations, for example for DMPs in a mixture of motor primitives [17], with Gaussian mixture models [18], and others. In our previous work we applied LWR [19] and GPR [20] for the generalization of DMPs. In contrast to these works, where statistical generalization has been performed on a set of position trajectories, we take into account also the forces and torques that arise in contact with the environment, when the peg is being inserted. This is the main contribution of the paper. The combination of generalization and subsequent adaptation provides us with a very versatile solution, which can be applied to different robotic platforms, be it humanoid or industrial robots.

This paper is organized as follows. In the next section we provide an overview of the proposed approach. Separate segments of the used method are explained in detail in Section III. Experimental evaluation follows in Section IV, with conclusions given in Section V.

II. Method overview

The method presented in this paper consists of three steps. In the first step, example motion patterns are recorded and stored in a library together with the associated external conditions. In the second step, generalized skill instances are computed using statistical approaches. The third step deals with the execution and optimization of the newly generated skill, which involves contact with the environment. Fig. 1 shows the flow chart of the proposed approach. A number of example trajectories need to be recorded to perform statistical generalization. As previously explained, in our approach not only movement trajectories but also forces and torques are taken into account. One of the most suitable methods for collecting the data is kinesthetic guiding, where the robot position and orientation trajectories combined with the resulting forces and torques can be recorded.

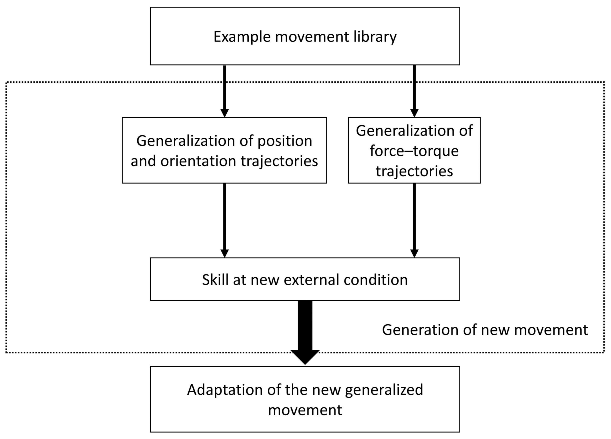
Example movement library
Generalization of position and orientation trajectories
Generalization of force–torque trajectories
Skill at new external condition
Generation of new movement
Adaptation of the new generalized movement

Fig. 1. Flow chart of the proposed method

Generalization of example movements from the library is performed with locally weighted regression. LWR is a memory-oriented, non-parametric method for statistical approximation [14]. The basic idea is to compute local models using data from a neighborhood of the desired query point. Since only a limited number of examples from the neighborhood of the current query are used to compute new movements, the method can be applied on-line. In our system, the output of the generalization algorithm are 1) parameterized position and orientation trajectories encoded as Dynamic Movement Primitives (DMPs), and 2) force-torque profiles encoded as a linear combination of Radial Basis Functions (RBF).

Newly obtained trajectories (DMPs) in tasks involving contacts cannot be executed directly by a robot because even small differences from the optimal trajectory can cause large forces and torques, which can result in a failed task execution. Hence, the generalized trajectories have to be properly adapted. In our work, we use a position controlled method based on admittance control that adapts the position and orientation trajectories based on differences between the generalized forces and torques and the actual forces and torques that arise during the execution. Upon this, an offset vector is calculated and added to the positional and orientational part of the executed trajectory, resulting in smaller discrepancies between the generalized and actual forces and torques thus, changing the trajectory during the execution.

III. Description of generalization method

A. Recording the example movement library

Robot task trajectories can be recorded in several ways. In this work, we focus on robot data recording with kinesthetic guiding [21], shown in Fig. 2. Movement trajectories are recorded in Cartesian coordinates together with end effector forces and torques. Note that some robots use joint torques to estimate the end effector forces and torques, therefore the

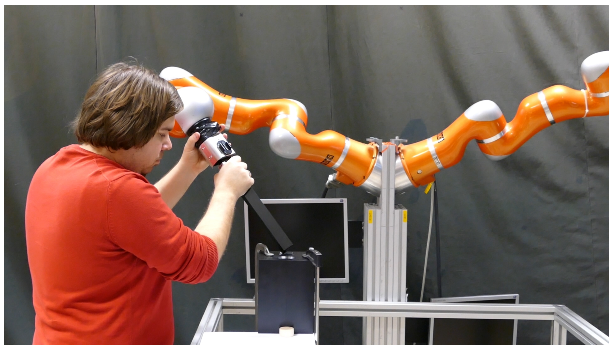

Fig. 2. Kinesthetic guiding of Kuka LWR-4 anthropomorphic robot arm, which was used to record the example movement library. The robot motion is first recorded and then reproduced to record also forces and torques.

measured forces can be corrupted by the human demonstrator. In such cases, net forces and torques are obtained with repeating the demonstrated motion, in exactly the same configuration of the workcell. In the recording phase, the operator physically guides the robot along the desired trajectory, and thus receives the same feedback from the environment as the robot. Consequently, the initial trajectories are more optimal and it is not necessary to refine them afterwards with means of reinforcement learning [8] or similar methods.

The following data is recorded when demonstrating force-based tasks: Cartesian space positions and orientations (represented as quaternions), their first and second derivatives (linear/angular velocities and accelerations)

$$G_d = \{\mathbf{p}_{ij}, \boldsymbol{q}_{ij}, \dot{\boldsymbol{p}}_{ij}, \boldsymbol{\omega}_{ij}, \ddot{\boldsymbol{p}}_{ij}, \dot{\boldsymbol{\omega}}_{ij}, t_{ij}\}_{j=1,\ i=1}^{T_i,\ NumEx}, \tag{1}$$

and the corresponding forces and torques

$$F_d = \{\boldsymbol{F}_{ij}, \boldsymbol{M}_{ij}, t_{ij}\}_{j=1,\ i=1}^{T_i,\ NumEx}, \tag{2}$$

all recorded at times t_{ij}, $j = 0, \ldots, T_i$. T_i is the number of samples in the i-th recording, $i = 1, \ldots, NumEx$, is the index of the recording, and $NumEx$ is the number of recordings (example task executions in the library). Orientations are given as unit quaternions $\boldsymbol{q} = (v, \boldsymbol{u}) \in \mathbb{R}^4$. Note that all PiH trajectories used for generalization were segmented according to the contact between the peg and the base object, discarding the approach movement from the recordings.

Trajectories and force/torque profiles are recorded at different external conditions (queries). These queries, which in our case are given as the depth of the hole[1] h_i, are also saved in the example library

$$H_d = \{h_i\}_{i=1}^{NumEx}. \tag{3}$$

Note that the distribution of the example trajectories and the corresponding queries influences the success of the generalization. Best results are obtained if the queries are evenly distributed. Besides, the recorded data must transition smoothly between queries, see also [20].

[1]The chosen external condition – the depth of the hole – emphasizes the advantage of generalizing the forces/torques in addition to the position trajectories. While the position/orientation trajectories are similar in this case, and therefore easy to generalize, the force/torques are qualitatively different – not, for example, just rotated.

B. Generating new movements from example library

We applied locally weighted regression (LWR) [14] for statistical generalization of the recorded data. LWR is a non-parametric method for statistical approximation, which uses raw trajectories and force/torque profiles stored in memory to determine new movements. This approach was used for generalization in [19], where LWR was used to generalize throwing, reaching and drumming movements without considering contacts with the environment. In contrast to this previous work, here we consider also the forces and torques that were recorded and stored in the example library.

In our system, Cartesian positions and orientations represented by unit quaternions are encoded as DMPs. As explained in the Appendix, DMPs have the following free parameters: weights $\mathbf{w} \in \mathbb{R}^{N\times 7}$, where N is the number of basis functions used to approximate the movement, trajectory duration τ and the final position and orientation of the robot $\mathbf{g} \in \mathbb{R}^7$. Note that each Cartesian position and each element of the quaternion is encoded as a separate DMP. After generalization, the orientational part of the trajectories have to be normalized to ensure the unit length of the quaternion, as generalization does not preserve the norm.

Force/torque profiles from the example library are not encoded by DMPs because in this case the first and the second derivative are not relevant. Instead we use a linear combination of radial basis functions in a form

$$\boldsymbol{F}(x) = \frac{\sum_{k=1}^{N} \boldsymbol{v}_k^F \Psi_k(x)}{\sum_{k=1}^{N} \Psi_k(x)} x, \tag{4}$$

$$\boldsymbol{M}(x) = \frac{\sum_{k=1}^{N} \boldsymbol{v}_k^M \Psi_k(x)}{\sum_{k=1}^{N} \Psi_k(x)} x. \tag{5}$$

Both position/orientation, encoded as DMPs, and forces/torques, encoded as RBF (analogous to DMP weight parameters), share the same phase variable x. The weights $\boldsymbol{v} = [v_1, \ldots, v_M]^T$ need to be computed to approximate each component of the desired force/torque profile.

The basic idea of our approach is to apply locally weighted regression to compute local models, using the data close to the desired query point

$$G_p(G_d, H_d) : h \to [\mathbf{w}^T, \tau, \boldsymbol{g}^T]^T \tag{6}$$

for positions and orientation trajectories and

$$G_f(F_d, H_d) : h \to \boldsymbol{v}, \tag{7}$$

for forces and torques. Here G_p and G_f are the generalization functions that map a given query point h into the desired positions and orientation trajectories and force/torque profiles, respectively.

The generation of generalization function G_p is the same as in [19] and the reader is related to this paper for implementation details. In the following we therefore explain only the generation of function G_f.

The computation of radial basis functions weights $\mathbf{v}$ is based on the training data set (2), (3). Lets first describe how to approximate the force profile of the i-th demonstration with a linear combination of RBFs. The following training data is available for learning:

$$F(x_{i,j}) = \frac{\sum_{k=1}^{N} v_k{}^F \Psi_k(x_{i,j})}{\sum_{k=1}^{N} \Psi_k(x_{i,j})} x_{i,j}, j = 0, \ldots, T_i, \tag{8}$$

where the phase x is defined by the corresponding DMP representing the position/orientation trajectory, thus it is given by $x_{i,j} = \exp(-\alpha_x t_{i,j}/\tau)$. (8) is a system of linear equations in v_i. Therefore it can be written in matrix form: $\boldsymbol{f}_i = \boldsymbol{X}_i \boldsymbol{v}$, with the system matrix $\boldsymbol{X}_i$ defined as

$$\boldsymbol{X}_i = \begin{bmatrix} \frac{\psi_1(x_{i,0}) x_{i,0}}{\sum_{k=1}^{N} \psi_k(x_{i,0})} & \cdots & \frac{\psi_N(x_{i,0}) x_{i,0}}{\sum_{k=1}^{N} \psi_k(x_{i,0})} \\ \vdots & \ddots & \vdots \\ \frac{\psi_1(x_{i,T_i}) x_{i,T_i}}{\sum_{k=1}^{N} \psi_l(x_{i,T_i})} & \cdots & \frac{\psi_N(x_{i,T_i}) x_{i,T_i}}{\sum_{k=1}^{N} \psi_k(x_{i,T_i})} \end{bmatrix} \tag{9}$$

and the left and right hand side vectors defined as

$$\boldsymbol{f}_i = \begin{bmatrix} F(x_{i,0}) \\ \vdots \\ F(x_{i,T_i}) \end{bmatrix}, \ \boldsymbol{v} = \begin{bmatrix} v_1^F \\ \vdots \\ v_N^F \end{bmatrix}. \tag{10}$$

Thus for the i-th demonstration we can calculate $\boldsymbol{v}$ using

$$\boldsymbol{v} = \boldsymbol{X}_i^+ \boldsymbol{f}_i, \tag{11}$$

where $\boldsymbol{X}_i^+$ denotes the Moore-Penrose pseudoinverse of $\boldsymbol{X}_i$.

By applying LWR we can generalize the training data G_d, H_d to a new query point, in our case new depth h, by solving the following least squares optimization problem

$$\min_{\boldsymbol{v}} \sum_{i=1}^{NumEx} \|\boldsymbol{X}_i \boldsymbol{v} - \boldsymbol{f}_i\|^2 K(h - h_i). \tag{12}$$

Here K is the kernel function that puts more emphasis on data associated with queries h_i closer to the current query h. There are many possible choices to select the weighting function K [14]. We chose the tricube kernel

$$K(d) = \begin{cases} (1 - |d|^3)^3 & \text{if } |d| < 1 \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

because this kernel has finite support. It also has continuous first and second derivatives, which makes the first two derivatives of the generalization function G_f continuous. By choosing the tricube kernel we also reduce the computational complexity of the optimization problem (12) because the force/torque profiles for which K vanishes do not influence G_f, which reduces the size of the system matrix associated with the objective function (12). As discussed in [14], the choice of the weighting kernel is rarely critical for the performance of locally weighted regression. We obtained good performance with the tricube kernel in our experiments.

The generalization is performed separately for each degree of freedom (using the method described in [19]) and each component of the force/torque profile (using the method described above). This results in 13 separate generalization functions (3 for position trajectory, 4 for orientation trajectories and 6 for force/torque profile).

C. Adaptation of generalized trajectories

While executing the generalized trajectories, the resulting forces and torques can differ from the ones calculated with the generalization method. If these discrepancies are significant, this could cause the task execution, in our case PiH execution, to fail or even damage the objects or the robot. Therefore, the generalized trajectories have to be adapted during the execution. As proposed in the work of Abu-Dakka et al. [6], an error feedback calculated from the actual and demonstrated forces and torques can be used to modify the robot movement, thus reducing the discrepancies between the desired and actual forces and torques. In our work, we apply this method with the modification of the error feedback. Instead of using the demonstrated forces and torques, the error feedback is calculated using the discrepancies between the *generalized* and actual forces-torques that arise during the execution. The error feedback $\boldsymbol{e}_p(x)$ for positions and $\boldsymbol{e}_q(x)$ for orientations can thus be calculated as

$$(0, \boldsymbol{e}_p(x)) = \boldsymbol{q}(x) * (0, \boldsymbol{F}_{gen}(x) - \boldsymbol{F}_{mes}) * \bar{\boldsymbol{q}}(x) \tag{14}$$

$$(0, \boldsymbol{e}_q(x)) = \boldsymbol{q}(x) * (0, \boldsymbol{M}_{gen}(x) - \boldsymbol{M}_{mes}) * \bar{\boldsymbol{q}}(x) \tag{15}$$

where $\boldsymbol{F}_{gen}$ and $\boldsymbol{M}_{gen}$ are the generalized force and torque at phase x, respectively, $\boldsymbol{F}_{mes}$ and $\boldsymbol{M}_{mes}$ the current measured force and torque, $\boldsymbol{q}(x)$ is the unit quaternion which specifies the current orientation of the robot's tool, and $*$ denotes the quaternion product. Using this error feedback, the trajectories can be modified as proposed in [6]. The trajectory adaptation algorithm based on admittance control uses the DMP phase stopping mechanism described in [6], [22]. If the discrepancies between the measured and the generalized forces increase, the phase needs to be slowed down, to prevent the robot from jamming. The method of Abu-Dakka et al. [6] utilizes the online feedback error to learn a feedforward term, which is used in execution of the task at the desired query point. This feedforward term makes the measured forces and torques closer to the generalized forces and torques, thereby reducing the need for slowing down/phase stopping and consequently increasing the speed of execution.

IV. Experimental Evaluation

The proposed method was implemented and tested on two different robot platforms:

- Kuka LWR-4 with 7 DOFs, equipped with an RH-707 two finger gripper. This robot has a torque sensor in every joint, therefore no external force/torque sensor is needed. The measured joint torques are transformed to Cartesian forces and torques using the robots dynamic model.
- Universal Robot arm – type UR5 – with 6 DOFs, equipped with ATI (Gamma SI-130-10) force/torque sensor located at the wrist of the robot and a SCHUNK WSG50 two finger gripper. This robot is driven by a high gain non-compliant controller. UR5 robot features active gravity compensation and free movement in teaching mode, thereby enabling kinesthetic guiding.

To evaluate the proposed approach, we performed a task of inserting both wooden and plastic pegs into holes, where the depth of the hole is the external condition of the task - the query for generalization. Note that the type of material and tolerances significantly affect the execution of the PiH task and change the resulting forces and torques. Besides, pegs and holes of round and square shape were used. Tolerance between the wooden peg (Fig. 11) and hole was 1.2 mm and the tolerance between both square and round plastic pegs and respective holes was 0.3 mm (Fig. 2). Example movements and force/torque profiles were obtained using kinesthetic guidance on both robots, as described in Section III-A. The total depth varied from 4 cm to 16 cm in 2 cm increments. With kinesthetic guidance we obtained 7 example trajectories, at depths of $16+[0,-2,-4,-6,-8,-10,-12]$ cm. Our method was evaluated for 6 previously unexplored movements at depths $16+[-1,-3,-5,-7,-9,-11]$ cm. New position trajectories and force/torque profiles were generated for these depths. The resulting forces during the execution can be different from the generalized forces and torques, therefore the trajectories were further refined using the approach from [6].

A. Experiments performed with Kuka LWR-4

The first set of experiments was conducted with Kuka LWR-4 robot arm inserting the plastic (ABS) round peg into the round hole. Robot control and the proposed method were implemented in Matlab, which communicated with the robot controller using Fast Research Interface (FRI) [23]. Although Kuka LWR robots have the capability of compliant control, in this experiments this capability was not exploited and it was set to maximal Cartesian stiffness in order to provide a better comparison with the results gathered on the UR5 robot that is admittance controlled. Figures 3-6 show the example and the generalized trajectories and force/torque profiles at the query point $h=13$ cm.

Fig. 7 shows the generalized forces (blue dotted line) and the actually measured forces (green solid line) arising during the execution of the generalized trajectory. In Fig. 8 the phase evolution during the execution of the generalized trajectory is shown. Whenever there is a significant difference between the measured and the generalized forces and torques, the phase is slowed down (using the phase stopping technique explained in [6]) according to the force/torque error, which is partially compensated by the force feedback control. Note that the phase stopping has occurred towards the end of execution, which resulted in a prolonged execution time of the trajectory.

In order to demonstrate the benefit of the proposed generalization of the force/torque profiles, we performed experiments, where the proposed technique was used only for generalization of position trajectories. Instead, the nearest force/torque profile in the example library with respect to the given query point, e. g., depth $h=13$ cm, was used. Note that the selected force/torque profile has to be mapped to the time interval of the generalized trajectory. The corresponding results are outlined in Figs. 7 and 8. Comparing the force

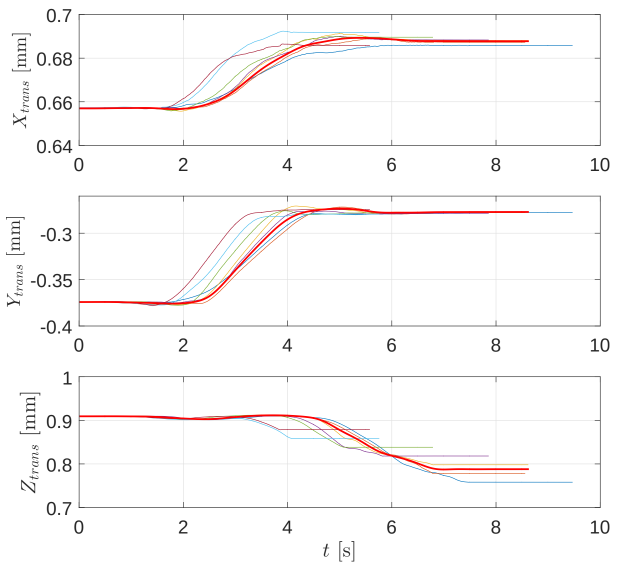

Fig. 3. Graphs show the Cartesian x, y, z coordinates of the PiH movement from the example movement library. Generalized position trajectories for the depth $h=13$ cm are shown in red.

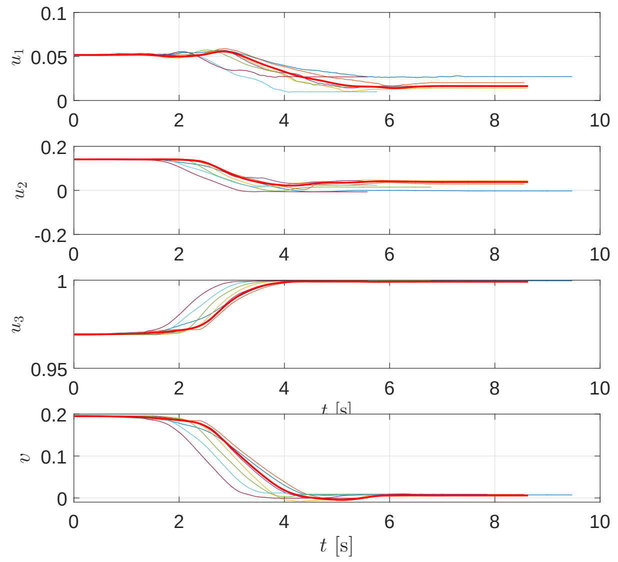

Fig. 4. Graphs show the quaternion orientations $q=(v,u)$ of the PiH movement saved in the example library. Generalized orientation trajectories for the depth $h=13$ cm are presented red.

evolution of the two executions in Fig. 7, we can see that the differences between the generalized forces (blue dotted line) and nearest neighbour forces (solid red line) are far greater than the differences between the measured forces, obtained with the execution of the generalized force/torque trajectory (green solid line) and the generalized forces. A significant difference can also be noted in phase evolution, which is shown in Fig. 8. The phase stopping was much more frequent in the execution with nearest neighbour forces compared, to the execution of measured force/torque trajectory throughout the entire evolution of the phase. This means that there was a major difference between the actual forces and the example forces, which resulted in a longer execution time. This experiment shows that generalization of force/torque profiles

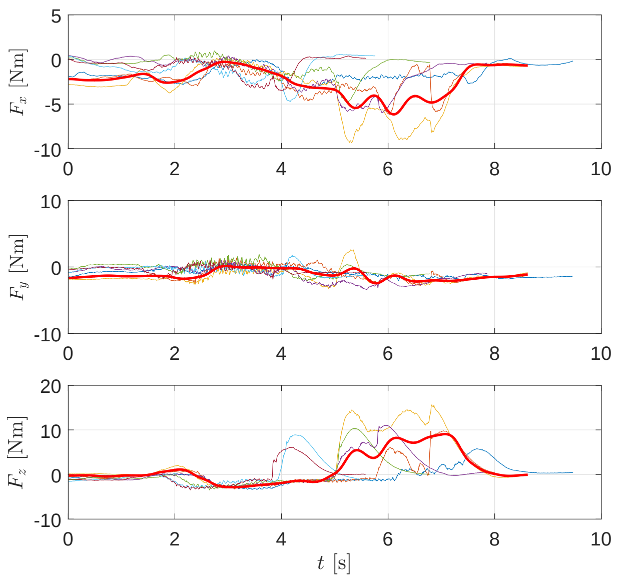

Fig. 5. Graphs show the Cartesian forces arising during the execution of the PiH task saved in the example library. Generalized forces for the depth $h = 13$ cm are shown in red.

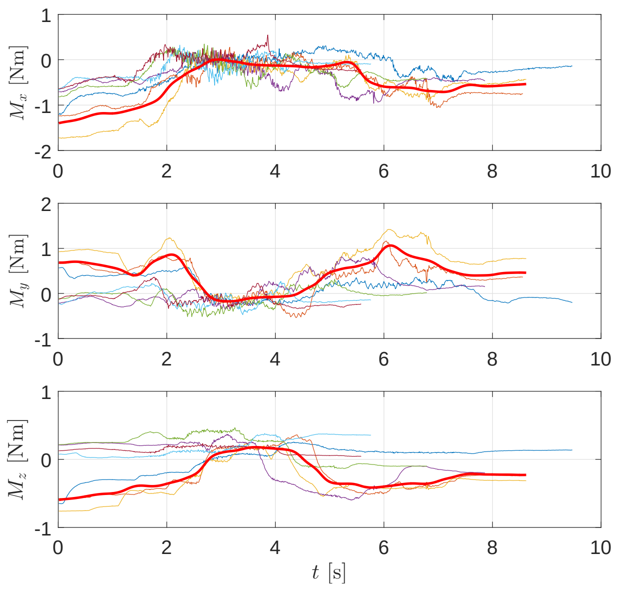

Fig. 6. Graphs illustrate the measured Cartesian torques of the PiH task from the example library. Generalized torques for the depth $h = 13$ cm are shown in red.

with LWR leads to a significantly faster task execution than the simple nearest neighbour method.

The same experimental setup was used to test the proposed approach on a square shaped peg and hole. First a set of example trajectories was recorded. Then, the same algorithm was used to generalize new movement trajectories with associated force/torque profiles, and executed. Comparison of nearest neighbour and generalized trajectory executions at depth $h = 13$ cm can be seen in Fig. 9 and Fig. 10. The comparison of graphs in Fig. 8 and Fig. 10 shows similar results, proving that the generalized forces require less adaptation for the PiH task execution. It can also be seen that the adaptation of the generalized trajectories with

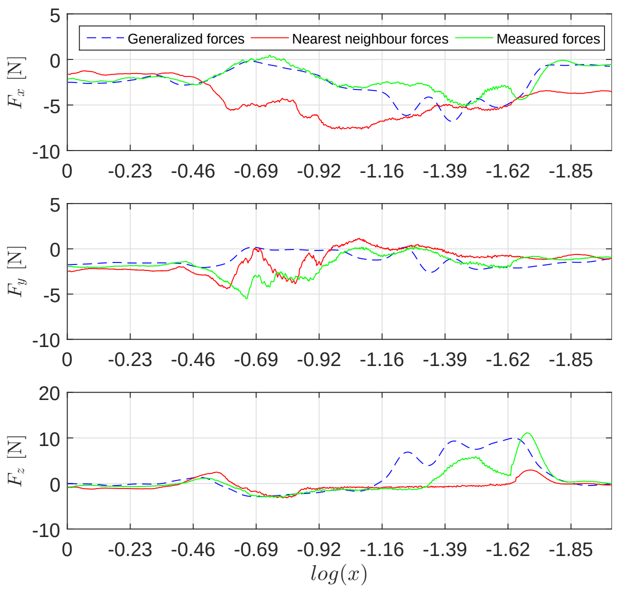

Fig. 7. Forces measured during the peg insertion. The dotted blue line represents the generalized forces. Solid red line represents the task execution with nearest neighbour forces and green line that represents the measured forces during the trajectory execution of generalized forces.

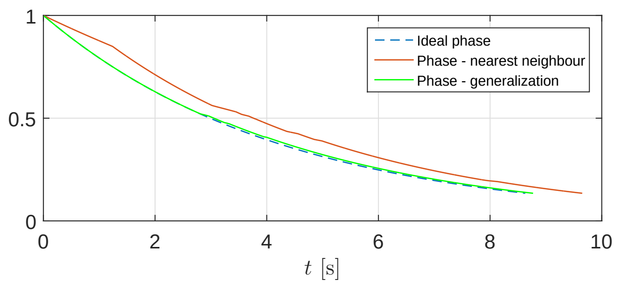

Fig. 8. Phase evolution during the execution of the generalized trajectory for the PiH task, using the measured force/torque profile and the nearest neighbour force/torque profile. The solid blue line corresponds to an ideal phase without DMP phase stopping, while the solid red line corresponds to the actual phase evolution during the nearest neighbour method execution. Green line represents the execution of the measured generalized force/torque trajectory.

the square peg insertion was more frequent comparing to the round peg. This is due to the shape of the object, because small misalignments and tight tolerances result in high forces exerted on the robot.

B. *Experimental evaluation with UR5 robot*

The proposed method was also tested on the Little Helper 4 platform at Aalborg University. The Universal Robot UR5 arm (shown in Fig. 11) is the main manipulator of the robotic platform. As before the algorithm was implemented in Matlab. ROS industrial driver was used as means of communication between the UR5 robot controller and Matlab with a frequency rate of 50 Hz.

The example trajectories were demonstrated using kinesthetic guidance. Unlike Kuka LWR-4, which is impedance controlled, UR5 is admittance controlled. Compared to kinesthetic guidance with Kuka LWR-4, kinesthetic guidance with UR5 is less smooth due to the limitations of sensors and control. Therefore, the wooden peg and a hole with a

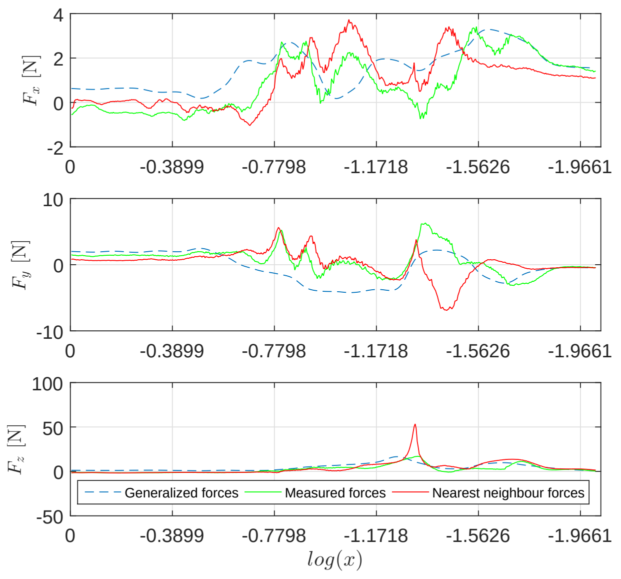

Fig. 9. Forces measured during the square peg insertion. The dashed blue line represents the generalized forces, solid red line represents the nearest neighbour forces. Measured forces during the generalized trajectory execution are represented with green solid line.

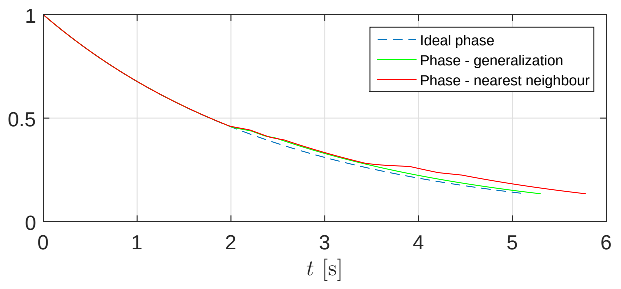

Fig. 10. Phase evolution with phase stoping during the generalized trajectory execution (ideal phase blue dashed line), using the measured force/torque profile (green) and the nearest neighbour force/torques (red).

tolerance of 1.2 mm were used. Figures 12 and 13 show the discrepancies between the generalized and measured forces and phase evolution of the executed generalized trajectory. The results are comparable with the results obtained with Kuka LWR-4, but the insertion is generally not as smooth and robust due to the admittance control performed at a low frequency rate.

V. Conclusions

We developed an approach for force-based statistical learning of contact skills from a set of example movements. In the proposed approach, the recorded forces and torques are taken into account besides position and orientation trajectories. The developed method generates new trajectories as well as the corresponding force/torque profiles from the demonstrated data. Position and orientation trajectories are encoded with DMPs, whereas force/torque profiles are encoded with RBFs. In our work we used LWR as a method for generating new trajectories according to a given query point.

In stiff environments generalized trajectories might still require additional adaptation in order to generate the desired force/torque profile. By applying a suitable trajectory adaptation method, the generalized trajectory can be modified so that the generalized forces and torques match the ones that arise during the task execution. We tested our method on two different robotic platforms: the LWR-4 humanoid arm, which is impedance controlled, and the UR-5 industrial robot arm, which is admittance controlled. Despite of different controllers the proposed method was successfully applied on both platforms. Experimental results show the robustness and fast adaptation capability of the proposed generalization and adaptation algorithm. In our future work we will investigate other relevant task external conditions in order to enhance the performance and speed up the execution.

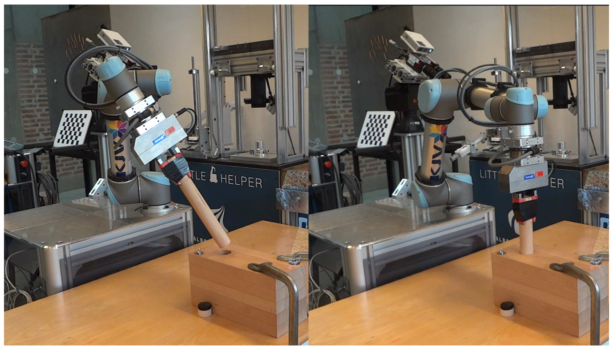

Fig. 11. Generalized trajectory execution on Universal Robot UR5.

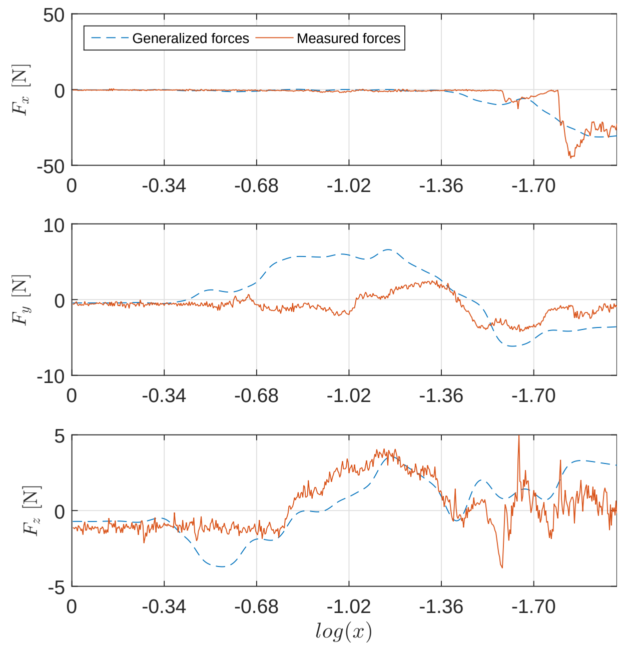

Fig. 12. Forces measured during the peg insertion with UR5. The reference generalized forces are represented with the blue dashed line and the actual measured forces with the red solid line.

Appendix

We use DMPs to encode Cartesian space position and orientation trajectories. A DMP is specified by a set of nonlinear differential equations with well-defined attractor dynamics [9]. For a single degree of freedom trajectory y,

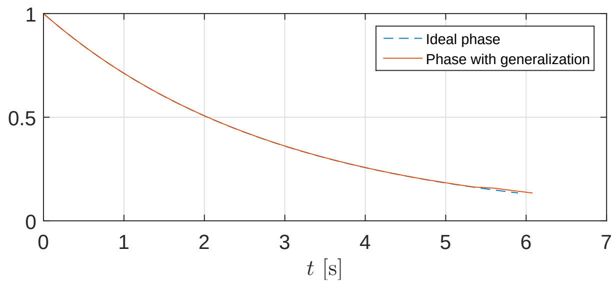

Fig. 13. Phase evolution of the peg insertion with UR5. Blue dashed line represents the phase of the generalized trajectory where no phase stopping is applied, while the red solid line shows the actual phase evolution of the executed trajectory, where phase stopping is applied to prevent jamming and enable adaptation. The nearest neighbor forces execution is not shown because such execution was not successful due to the required amount of adaptation, which was not feasible at the given frequency rate. Too high forces would appear and the robot would stop for safety reasons.

the DMP is defined as follows:

$$\begin{aligned} \tau\dot{z} &= \alpha_z(\beta_z(g-y)-z)+f(x), && (16)\\ \tau\dot{y} &= z, && (17)\\ \tau\dot{x} &= -\alpha_x x, && (18) \end{aligned}$$

where x is the phase variable, z is the auxiliary variable and g is the desired goal of the movement. Parameters α_z, β_z, α_x and τ define the behaviour of this second order system. If the parameters are selected as $\tau > 0$, $\alpha_z = 4\beta_z > 0$ and $\alpha_x > 0$, then the dynamic system has a unique point attractor at $y = g,\ z = 0$. Given the initial condition $x(0) = 1$, the Eq. (18) is solved analytically by $x(t) = \exp(-\alpha_x t/\tau)$. However, to implement different modulations of a DMP such as phase stopping [22], it is better to keep Eq. (18) as differential equation.

The forcing term $f(x)$ is defined as a linear combination of radial basis functions, which enable the robot to follow any smooth point-to-point trajectory from the beginning of the movement y_0 to the end configuration g:

$$\begin{aligned} f(x) &= \frac{\sum_{i=1}^{N} w_i\Psi_i(x)}{\sum_{i=1}^{N}\Psi_i(x)}x, && (19)\\ \Psi_i(x) &= \exp\left(-h_i\,(x-c_i)^2\right). && (20) \end{aligned}$$

Here c_i are the centers of the radial basis functions distributed along the trajectory and $h_i > 0$. For robots with more than one degree of freedom, each degree is represented by Eq. (16) – (17) with different w_i, and g, but with a common phase variable x and time constant τ as specified in Eq. (18).

To approximate any smooth trajectory with a DMP, we need to estimate the weights w_i, time constant τ, and the goal configuration g. τ is usually set to the duration of the movement, g to the final configuration on the trajectory, while w_i are estimated from the training data (sampled positions, velocities and accelerations) using regression techniques. See [9] for more details.

References

[1] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[2] R. Dillmann, "Teaching and learning of robot tasks via observation of human performance," *Robotics and Autonomous Systems*, vol. 47, no. 2, pp. 109–116, 2004.

[3] J. J. Steil, C. Emmerich, A. Swadzba, R. Grünberg, A. Nordmann, and S. Wrede, "Kinesthetic teaching using assisted gravity compensation for model-free trajectory generation in confined spaces," in *Gearing Up and Accelerating Cross-fertilization between Academic and Industrial Robotics Research in Europe:*, pp. 107–127, Springer, 2014.

[4] P. Kormushev, S. Calinon, and D. G. Caldwell, "Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input," *Advanced Robotics*, vol. 25, no. 5, pp. 581–603, 2011.

[5] F. Steinmetz, A. Montebelli, and V. Kyrki, "Simultaneous kinesthetic teaching of positional and force requirements for sequential in-contact tasks," in *IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pp. 202–209, 2015.

[6] F. J. Abu-Dakka, B. Nemec, J. A. Jørgensen, T. R. Savarimuthu, N. Krüger, and A. Ude, "Adaptation of manipulation skills in physical contact with the environment to reference force profiles," *Autonomous Robots*, vol. 39, no. 2, pp. 199–217, 2015.

[7] L. Rozo, P. Jiménez, and C. Torras, "A robot learning from demonstration framework to perform force-based manipulation tasks," *Intelligent Service Robotics*, vol. 6, no. 1, pp. 33–51, 2013.

[8] M. Kalakrishnan, L. Righetti, P. Pastor, and S. Schaal, "Learning force control policies for compliant manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (San Francisco), pp. 4639–4644, 2011.

[9] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," *Neural computation*, vol. 25, no. 2, pp. 328–373, 2013.

[10] S. Schaal, P. Mohajerian, and A. Ijspeert, "Dynamics systems vs. optimal control - a unifying view," *Progress in brain research*, vol. 165, pp. 425–445, 2007.

[11] P. Pastor, L. Righetti, M. Kalakrishnan, and S. Schaal, "Online movement adaptation based on previous sensor experiences," in *International Conference on Intelligent Robots and Systems (IROS)*, pp. 365–371, 2011.

[12] A. Gams, B. Nemec, A. Ijspeert, and A. Ude, "Coupling movement primitives: Interaction with the environment and bimanual tasks," *IEEE Transactions on Robotics*, vol. 30, pp. 816–830, Aug 2014.

[13] V. Koropouli, S. Hirche, and D. Lee, "Generalization of force control policies from demonstrations for constrained robotic motion tasks," *Journal of Intelligent & Robotic Systems*, 2015. doi: 10.1007/s10846-015-0218-y.

[14] C. G. Atkeson, A. W. Moore, and S. Schaal, "Locally Weighted Learning," *Artificial Intelligence Review*, vol. 11, no. 1-5, pp. 11–73, 1997.

[15] S. Vijayakumar, A. D'Souza, T. Shibata, J. Conradt, and S. Schaal, "Statistical Learning for Humanoid Robots," *Autonomous Robots*, vol. 12, no. 1, pp. 55–69, 2002.

[16] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, Massachusetts: The MIT Press, 2006.

[17] K. Muelling, J. Kober, and J. Peters, "Learning table tennis with a mixture of motor primitives," in *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pp. 411–416, Dec 2010.

[18] S. Calinon, "Robot learning with task-parameterized generative models," in *Proc. Intl Symp. on Robotics Research*, Sept. 2015.

[19] 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.

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

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

[22] A. Ude, B. Nemec, T. Petrič, and J. Morimoto, "Orientation in Cartesian space dynamic movement primitives," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Hong Kong), pp. 2997–3004, 2014.

[23] G. Schreiber, A. Stemmer, and R. Bischoff, "The fast research interface for the kuka lightweight robot," in *IEEE Workshop on Innovative Robot Control Architectures for Demanding (Research) Applications How to Modify and Enhance Commercial Controllers (ICRA 2010)*, 2010.