Autonomous Robots manuscript No.
(will be inserted by the editor)

Adaptation of Manipulation Skills in Physical Contact with the Environment to Reference Force Profiles

Fares J. Abu-Dakka* · **Bojan Nemec*** · **Jimmy A. Jørgensen+** · **Thiusius R. Savarimuthu+** · **Norbert Krüger+** · **Aleš Ude***

Received: date / Accepted: date

Abstract We propose a new methodology for learning and adaption of manipulation skills that involve physical contact with the environment. Pure position control is unsuitable for such tasks because even small errors in the desired trajectory can cause significant deviations from the desired forces and torques. The proposed algorithm takes a reference Cartesian trajectory and force/torque profile as input and adapts the movement so that the resulting forces and torques match the reference profiles. The learning algorithm is based on dynamic movement primitives and quaternion representation of orientation, which provide a mathematical machinery for efficient and stable adaptation. Experimentally we show that the robot's performance can be significantly improved within a few iteration steps, compensating for vision and other errors that might arise during the execution of the task. We also show that our methodology is suitable both for robots with admittance and for robots with impedance control.

Keywords Skill Learning and Adaptation · Manipulation and Compliant Assembly

1 Introduction

Reliable transfer of skills involving physical interaction with the environment to new configurations is a difficult problem because even small errors in transfer parameters can cause failure. It is therefore necessary to provide methodologies that enable on-line adaptation based on the results of force feedback control. A suitable learning mechanism is required to improve the execution of the task if it has to be performed several times in the same configuration.

As a practical example we study typical operations in the automatic assembly, e. g. peg insertion, also referred to as Peg-in-Hole task (PiH), and extensions of PiH. Unavoidable positioning errors and tight tolerances between the objects involved in the PiH operation require compliance and on-line adaptation of the desired trajectories. Several approaches to solve the PiH problem based on force feedback control were proposed in the literature. Some of them focus on Remote-Center-of-Compliance (RCC), which is implemented as a passive mechanical device mounted on the wrist to prevent a peg from jamming when inserted into a hole with tight clearance (Laurin-Kovitz et al, 1991; Whitney and Nevins, 1979; Yun, 2008). Since RCC is a passive device, there are no problems with the stability of force control. Adaptation to the sensed forces can also be achieved by active control (Raibert and Craig, 1981), thus robotic controlled impedance devices that can be mounted on standard industrial robots have also been developed (Lopes and Almeida, 2008). Many active force control approaches have been proposed in the literature to solve the PiH task also with robot manipulators without auxiliary devices (Broenink and Tiernego, 1996; Hirana et al, 2002; Li, 1997; Stemmer et al, 2007; Yamashita et al, 1991). Active approaches can accommodate more complex assembly cases and larger positioning uncertainties (Newman et al, 1999). Both admittance (Gullapalli et al, 1992) and impedance control (Broenink and Tiernego, 1996) have been studied in this context. Recent studies have shown that active force control is required also for grasping in conjunction with dextrous grippers (Kazemi et al, 2014).

*Humanoid and Cognitive Robotics Lab, Dept. of Automatics, Biocybernetics and Robotics, Jožef Stefan Institute, Ljubljana, Slovenia.
E-mail: ales.ude@ijs.si ·
+Cognitive Vision Lab, Maersk Mc-Kinney Moller Institute, University of Southern Denmark, Odense

Since the appropriate strategy depends on the geometry of the manipulated workpiece, often an engineer has to hardcode a different strategy for every new workpiece geometry (Bruyninckx et al, 1995; Xiao, 1997), although automatic extraction of assembly plans is possible in some cases when CAD models are available (Stemmer et al, 2007). Besides force sensing, visual servoing and reactive task controllers can also play an important role (Hamner et al, 2010).

While active compliance can deal with more complex assembly cases and larger positioning uncertainties than passive compliance, it still cannot reach human performance (Giordano et al, 2008). Although force control can be fast as evidenced by research on legged locomotion (Hutter et al, 2012; Hyon et al, 2007), attempts to speed up robots with algorithms that rely on active force control has resulted in contact instability in assembly tasks (Newman et al, 1999). Problems with reliability when high gain force control is used have been confirmed in our own initial experiments, where jamming often occurred when the peg was inserted into a hole. This gives rise to an idea to first execute the movement slower and then gradually increase the speed of execution through learning.

Here we would like to point out that while in this paper we focus on automatic assembly tasks, the proposed algorithm is more general and can accommodate also other tasks that involve contact with the environment, e. g. force-based trajectory following. Our general approach to learning force-based skills consists of two stages

1. Acquire a reference trajectory and force/torque profile that successfully solve the task in a specific workspace configuration. We use programming by demonstration (PbD) for this purpose.
2. Transfer the acquired skill to new configurations using 3-D vision and adapt the resulting motion to the trained force/torque pattern.

Since humans are very good at performing assembly tasks that require compliance and force control, we use human demonstration of the task as a starting point. Unlike standard PbD approaches (Dillmann, 2004), we use besides trajectories also forces and torques arising during the task execution as training data. Previous works in this area include (Calinon et al, 2009; Kormushev et al, 2011; Rozo et al, 2013), who used haptic interfaces to demonstrate the training trajectories and the associated forces and torques. These authors focus on how to encode the demonstrations with Gaussian mixture models to reproduce the desired task, where the positions and forces are encoded in one model. This is different from our approach where the demonstrated trajectories are modified to achieve the desired force/torque profile. (Kaiser and Dillmann, 1996) also studied force-based skill learning but compute motor commands directly from force response without considering the phase and orientation, which is a problem for general PiH-like tasks. The work of (Skubic and Volz, 1998) focuses on discrete force-based events to select an appropriate controller, which is insufficient for PiH-like tasks that require continuous force control.

The main focus of this paper is on item 2, i. e. the refinement and transfer of known manipulation skills to new configurations of the robot's workspace. Consider for example the PiH task. If a new workpiece pose is determined by vision, the robot cannot simply translate and rotate the previously trained Cartesian space trajectory and replay it. Such an approach is bound to fail due to noise in the estimated workpiece pose, uncertainties in the posture of the peg in the gripper, and due to a different joint space configuration of the robot arm, which can cause Cartesian space tracking errors (Gullapalli et al, 1992). Instead we attempt to improve the transformed control policy by matching it with the previously trained performance in the force/torque space using force feedback control. The main idea is to adapt the transformed Cartesian space trajectory on-line so that the resulting forces and torques match the recorded force/torque profile. If the task is repeated several times, our learning procedure moves the force feedback errors to an offset trajectory, thus improving the execution of the task and making it closer in speed to the originally trained performance. Our experimental results show that with the proposed approach a robot can effectively learn and transfer assembly operations to new situations. A graphical illustration of the approach is provided in Section 3.4.

We selected Dynamic Movement Primitives (DMPs) (Ijspeert et al, 2001, 2013) to encode and refine the demonstrated movements. The motivation for this was that DMPs can be effectively used to encode and reproduce the demonstrated movements. In addition, the mechanism of phase stopping as introduced in (Ijspeert et al, 2001) can be used to slow down the robot execution, which prevents the jamming problem mentioned above that can occur in the presence of large uncertainties.

The paper consists of five sections. In Section 2 we briefly present the data acquisition procedure used for acquiring trajectories and force/torque profiles for assembly operations. In Section 3 we start by formulating the problem and introducing the dynamic movement primitives framework for discrete Cartesian space trajectories, including the mathematical machinery that enables quaternion based orientation control and phase stopping. We then present an approach to iteratively

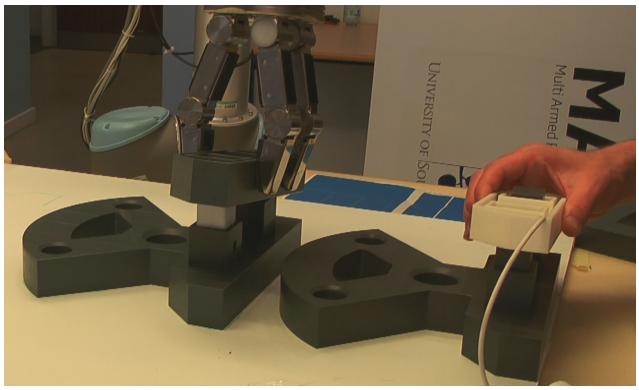


Fig. 1 Teleoperation setup with the position tracker inserted into the object and the Universal Robot arm.

Fig. 2 Kinesthetic guiding using Kuka LWR arm in gravity compensation mode.

adapt the learned trajectory to improve the task performance using force feedback. This procedure exploits the quaternion based control and phase stopping. Results of the experimental evaluation on two different platforms, one with admittance and one with impedance control, are presented in Section 4. The main contributions are summarized in the final section.

2 Acquisition of Training Data for Force-Based Manipulation Skills

We first describe two example procedures for learning of operations that involve contact with the environment by human demonstration. We record human demonstrations with two different systems:

- Teleoperation using a TrackSTAR 6D pose tracker from Ascension attached to the object (Fig. 1).
- Kinesthetic guiding using the Kuka LWR arm in gravity compensation mode (Fig. 2).

In the first setup we use a magnetic tracker attached to the object, which is held by a human demonstrator, to track the object position and orientation. The demonstrator executes the task, e. g. peg-in-hole, with his own hand. The measured poses are used to teleoperate the robot after being translated and rotated in such a way that the peg held by the robot is in the hole when the peg held by a human is in the hole. This is a constant transformation, which is estimated by the magnetic tracker in an initial calibration phase. The developed demonstration system is precise enough so that the teleoperated robot successfully executes the task together with a human (see also the attached video). More details about this systems are given in (Savarimuthu et al, 2013).

The second setup uses the robot's gravity compensation mode to enable kinesthetic guiding (Hersch et al, 2008; Lee and Ott, 2011), where a human operator guides the robot's tool center point along the desired trajectory in such a way that the desired task is successfully executed. We measure the resulting joint space motion by proprioception. In Kuka LWR arm, the torque sensors are located in the robot's joints and consequently, the forces exerted by the human operator during the demonstration affect the measured joint torques. Therefore, to get the net forces and torques at the robot tool center point without the influence of forces exerted by the human operator, we replay the measured joint space trajectory and record the resulting Cartesian space trajectory and joint torques. The measured joint torques are transformed into the corresponding tool center point forces and torques. We take care that the workspace configuration does not change during this process and since our robot can accurately track the joint space trajectories, the newly measured forces and torques provide a good reference for adaptation later. To avoid replaying the recorded trajectory, the procedure for estimating the forces and torques associated with the previously recorded trajectory could exploit a force/torque sensor mounted at the robot wrist and measuring the resulting forces and torques with this sensor.

The data acquired by both of our systems can be improved by means of reinforcement learning (Kalakrishnan et al, 2011), where a suitable cost function is minimized. The cost function should relate to the desired properties of the task, e. g. smoothness of execution, execution time, etc. Note that since human does not feel the same forces and torques as the robot in our first demonstration setup, the recorded force/torque profiles can sometimes be significantly improved through reinforcement learning, e. g. using the approach of (Kalakrishnan et al, 2011). On the other hand, in our second setup the initial trajectories are more optimal since the human directly guides the robot and thus receives the same feedback from the environment as the robot. Con-

sequently, refinement through reinforcement learning is normally not necessary with this system.

With both systems for the training of force-based manipulation skills, we acquire the Cartesian space tool center point positions and orientations (represented as quaternions), angular velocities and accelerations

$$\mathcal{G}_d = \{\mathbf{p}_j, \mathbf{q}_j, \dot{\mathbf{p}}_j, \boldsymbol{\omega}_j, \ddot{\mathbf{p}}_j, \dot{\boldsymbol{\omega}}_j\}_{j=0}^{T}, \tag{1}$$

and the associated Cartesian space forces and torques specified in hand coordinate frame

$$\mathcal{F}_d = \{\mathbf{F}_j, \mathbf{M}_j\}_{j=0}^{T}, \tag{2}$$

all acquired at times t_j, $j = 0, \ldots, T$. Positions and orientations are represented as a frame $\mathcal{Q} = \{\mathbf{p}, \mathbf{q}\}$, where $\mathbf{p} \in \mathbb{R}^3$ and $\mathbf{q} = v + \mathbf{u}$, $v \in \mathbb{R}$, $\mathbf{u} \in \mathbb{R}^3$, is a unit quaternion. We denote the space of all unit quaternions, which forms a unit sphere in $\mathbb{R}^4$, by S^3. The rest of this paper deals with how such training data can be used for learning and adaptation to new situations.

3 Policy Learning and Adaptation

The acquired Cartesian space trajectories that result in a successful execution of the task are first encoded with dynamic movement primitives (DMPs). A DMP for a single degree of freedom trajectory y is defined by the following set of nonlinear differential equations (Ijspeert et al, 2013)

$$\tau \dot{z} = \alpha_z(\beta_z(g - y) - z) + f(x), \tag{3}$$

$$\tau \dot{y} = z, \tag{4}$$

$$\tau \dot{x} = -\alpha_x x, \tag{5}$$

where x is the phase variable and z is an auxiliary variable. Parameters α_z and β_z define the behavior of the second order system described by Eq. (3) and (4). With the choice $\tau > 0$, $\alpha_z = 4\beta_z$ and $\alpha_x > 0$, the convergence of the underlying dynamic system to a unique attractor point at $y = g$, $z = 0$ is ensured (Ijspeert et al, 2013). $f(x)$ is defined as a linear combination of N nonlinear radial basis functions, which enables the robot to follow any smooth trajectory from the initial position y_0 to the final configuration g

$$f(x) = \frac{\sum_{i=1}^{N} w_i \Psi_i(x)}{\sum_{i=1}^{N} \Psi_i(x)} x(g - y_0), \tag{6}$$

$$\Psi_i(x) = \exp\left(-h_i \left(x - c_i\right)^2\right), \tag{7}$$

where c_i are the centers of Gaussians distributed along the phase of the trajectory and h_i are their widths. For a given N and setting the time constant $\tau = t_T$, where t_T is the time duration of the movement, we can define $c_i = \exp\left(-\alpha_x \frac{i-1}{N-1}\right)$, $h_i = \frac{1}{(c_{i+1} - c_i)^2}$, $h_N = h_{N-1}$, $i = 1, \ldots, N$. For each Cartesian degree of freedom, the weights w_i are estimated from the measured data (1) using regression (Ude et al, 2010), while g is the last recorded configuration on the trajectory. In our experiments we used $N = 20$, while other constants were set to $\alpha_z = 48$, $\beta_z = 12$, and $\alpha_x = 2$. For the sequencing of DMPs, e. g. to smoothly transition from the approach to the peg insertion movement, we use the methodology proposed in (Nemec and Ude, 2012). In the case of PiH task, there are two separate phases: the approach phase before the contact with the workpiece has been established, and the insertion phase, where the peg is inserted into the appropriate hole. Since our algorithm is suitable for force-based tasks, it is applied only to the second part of the movement, i. e. the insertion phase.

3.1 Cartesian Space DMPs

In the basic DMP equations (3) – (4), each degree of freedom is encoded as a separate DMP, but with the common phase variable x. However, direct integration of unit quaternions, which we use to represent 3-D orientation, does not ensure that the norm of quaternions stays equal to 1. Any representation of orientation that does not contain singularities is non-minimal, which means that additional constraints need to be taken into account during integration. (Pastor et al, 2011) proposed an integration step that ensures that the unity norm of quaternions is preserved during integration. We rewrite their quaternion DMP formalism in the original DMP form (3) – (4) and modify it by applying a logarithmic map to better account for angular velocities

$$\tau \dot{\boldsymbol{\eta}} = \alpha_z \left(\beta_z 2 \log\left(\mathbf{g}_o * \overline{\mathbf{q}}\right) - \boldsymbol{\eta}\right) + \mathbf{f}_o(x), \tag{8}$$

$$\tau \dot{\mathbf{q}} = \frac{1}{2} \boldsymbol{\eta} * \mathbf{q}, \tag{9}$$

where $\mathbf{g}_o \in \mathrm{S}^3$ denotes the goal orientation, bar denotes the quaternion conjugation defined as $\overline{\mathbf{q}} = \overline{v + \mathbf{u}} = v - \mathbf{u}$ and $*$ denotes the quaternion product defined as

$$\begin{aligned} \mathbf{q}_1 * \mathbf{q}_2 &= (v_1 + \mathbf{u}_1) * (v_2 + \mathbf{u}_2) \\ &= (v_1 v_2 - \mathbf{u}_1^{\mathrm{T}} \mathbf{u}_2) + (v_1 \mathbf{u}_2 + v_2 \mathbf{u}_1 + \mathbf{u}_1 \times \mathbf{u}_2). \end{aligned} \tag{10}$$

$\boldsymbol{\eta} \in \mathbb{R}^3$ is treated as quaternion with zero scalar part in (9). The quaternion logarithm $\log : \mathrm{S}^3 \mapsto \mathbb{R}^3$ is given as

$$\log(\mathbf{q}) = \log(v + \mathbf{u}) = \begin{cases} \arccos(v) \dfrac{\mathbf{u}}{\|\mathbf{u}\|}, & \mathbf{u} \neq 0 \\ [0, 0, 0]^{\mathrm{T}}, & \text{otherwise} \end{cases}. \tag{11}$$

It can be used to specify the distance metric on the space of unit quaternions S^3 (Ude, 1999)

$$\mathrm{d}(\mathbf{q}_1, \mathbf{q}_2) = \begin{cases} \| \log(\mathbf{q}_1 * \overline{\mathbf{q}}_2) \|, & \mathbf{q}_1 * \overline{\mathbf{q}}_2 \neq -1 + [0, 0, 0]^{\mathrm{T}} \\ \pi, & \text{otherwise} \end{cases}.$$

The motivation for the term $2\log(\mathbf{g}_o * \overline{\mathbf{q}})$ in (8) is the following relationship between the quaternion logarithm and the angular velocity

$$\boldsymbol{\omega} = 2\log(\mathbf{g}_o * \overline{\mathbf{q}}), \tag{12}$$

where $\boldsymbol{\omega}$ denotes the angular velocity that rotates quaternion $\mathbf{q}$ into $\mathbf{g}_o$ within unit sampling time. Thus only the application of the logarithmic map provides a proper mapping of the quaternion difference $\mathbf{g}_o * \overline{\mathbf{q}}$ onto the angular velocity.

The logarithmic map becomes one-to-one and continuously differentiable if we limit its domain to $\mathrm{S}^3/(-1+[0,0,0]^{\mathrm{T}})$. Thus in this case we can define its inverse, i. .e. the exponential map $\exp : \mathbb{R}^3 \mapsto \mathrm{S}^3$, as

$$\exp(\mathbf{r}) = \begin{cases} \cos(\|\mathbf{r}\|) + \sin(\|\mathbf{r}\|)\dfrac{\mathbf{r}}{\|\mathbf{r}\|}, \ \mathbf{r} \neq 0 \\ 1 + [0,0,0]^{\mathrm{T}}, \ \text{otherwise} \end{cases}. \tag{13}$$

Here the domain of the exponential map is limited to to $\|\mathbf{r}\| < \pi$. To simplify notation we also define the mapping χ, which denotes the transformation that maps an angular velocity into a unit quaternion describing the resulting rotation within time period Δt,

$$\chi(\Delta t\,\boldsymbol{\omega}) = \exp\left(\frac{\Delta t\,\boldsymbol{\omega}}{2}\right). \tag{14}$$

The nonlinear forcing term

$$\mathbf{f}_o(x) = \mathbf{D}_o \frac{\sum_{i=1}^{N} \mathbf{w}_i^o \Psi_i(x)}{\sum_{i=1}^{N} \Psi_i(x)} x \tag{15}$$

contains free parameters $\mathbf{w}_i^o \in \mathbb{R}^3$, which are used to encode any given orientation trajectory $\{\mathbf{q}_j, \boldsymbol{\omega}_j, \dot{\boldsymbol{\omega}}_j\}_{j=0}^{T}$ with the quaternion DMP specified by (8) – (9). The scaling factor $\mathbf{D}_o = \mathrm{diag}(\log(\mathbf{g}_o * \overline{\mathbf{q}}_0)) \in \mathbb{R}^{3\times 3}$ is a diagonal matrix built from a 3-D vector. The free parameters $\mathbf{w}_i^o$ are calculated by solving the following system of linear equations

$$\frac{\sum_{i=1}^{N} \mathbf{w}_i^o \Psi_i(x_j)}{\sum_{i=1}^{N} \Psi_i(x_j)} x_j = \mathbf{D}_o^{-1}\left(\tau\,\dot{\boldsymbol{\eta}}_j - \alpha_z\left(\beta_z\left(2\log\left(\mathbf{g}_o * \overline{\mathbf{q}}_j\right)\right) - \boldsymbol{\eta}_j\right)\right), \tag{16}$$

where phases x_j are obtained by integrating (5), i. e.

$$x_j = x(t_j) = \exp\left(-\frac{\alpha_x}{\tau} t_j\right). \tag{17}$$

and $j = 0, \ldots, T$. The relation between the quaternion derivative and angular velocity is given by $\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\omega} * \mathbf{q}$, thus from (9) we obtain $\boldsymbol{\eta}_j = \tau\,\boldsymbol{\omega}_j$, $\dot{\boldsymbol{\eta}}_j = \tau\,\dot{\boldsymbol{\omega}}_j$.

Unlike in our work, (Pastor et al, 2011) used in Eq. (8) just the vector part of the quaternion product $\mathbf{g}_o * \overline{\mathbf{q}}$ instead of $2\log(\mathbf{g}_o * \overline{\mathbf{q}})$

$$\tau\dot{\boldsymbol{\eta}} = \alpha_z\left(\beta_z \mathbf{g}_o * \overline{\mathbf{q}} - \boldsymbol{\eta}\right) + \mathbf{f}_o(x), \tag{18}$$

While this is equivalent as far as the direction of change is concerned, such a formulation does not properly relate to the angular velocity of robot movement. As shown in Eq. (12), it is necessary to apply the logarithmic map to transform the difference vector into angular velocity. Fig. 3 and 4 demonstrate that the proposed approach generates a much quicker response and therefore converges to the attractor point faster, which is the desired characteristics of the linear part of the DMP system. The DMP system with quaternion difference vector does not even come close to the attractor point in the time when the proposed system (8) – (9) has already converged. To make the system (18) closer to ours, we need to multiply the quaternion difference vector by 2

$$\tau\dot{\boldsymbol{\eta}} = \alpha_z\left(\beta_z 2\mathbf{g}_o * \overline{\mathbf{q}} - \boldsymbol{\eta}\right) + \mathbf{f}_o(x), \tag{19}$$

In this case the response of the DMP system (19), (9) is more similar to the proposed system with the logarithmic map, but it remains significantly slower. Note that 2, as evident from (12), is the correct multiplier to obtain a critically damped system in the DMP equations (8) – (9). If we further increase the multiplier, the resulting system is not critically damped but starts oscillating towards the goal orientation, which is suboptimal for robot control.

To simplify notation we also rewrite DMP equations (3) – (4), which are used to encode the positional part of the trajectory, in vector form

$$\tau\dot{\mathbf{z}} = \alpha_z(\beta_z(\mathbf{g}_p - \mathbf{p}) - \mathbf{z}) + \mathbf{f}_p(x), \tag{20}$$

$$\tau\dot{\mathbf{p}} = \mathbf{z}, \tag{21}$$

where $\mathbf{g}_p \in \mathbb{R}^3$ denotes the final position on the recorded trajectory. The forcing term $\mathbf{f}_p$ is defined as

$$\mathbf{f}_p(x) = \mathbf{D}_p \frac{\sum_{i=1}^{N} \mathbf{w}_i^p \Psi_i(x)}{\sum_{i=1}^{N} \Psi_i(x)} x, \tag{22}$$

where $\mathbf{D}_p = \mathrm{diag}(\mathbf{g}_p - \mathbf{p}_0) \in \mathbb{R}^{3\times 3}$. The diagonal matrices $\mathbf{D}_p$ in (22) and $\mathbf{D}_o$ in (15) are used to scale the movement amplitude if the goal configuration $\mathbf{g}_p$ and/or $\mathbf{g}_o$ change. The ability to scale orientation trajectories is another advantage of our approach compared to the quaternion DMP formulation of (Pastor et al, 2011). To track the desired Cartesian space trajectories, we need to integrate (20) – (21) and (8) – (9) together with the common phase (5). Euler integration can be used for this purpose except in the case of Eq. (9), where according to (14) the formula

$$\mathbf{q}(t + \Delta t) = \chi\left(\Delta t\,\frac{\boldsymbol{\eta}(t)}{\tau}\right) * \mathbf{q}(t) \tag{23}$$

should be used instead.

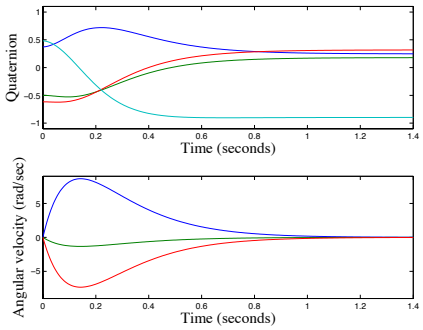


Fig. 3 Response of DMP system (8) – (9) without nonlinear term $\mathbf{f}_o$ (this paper).

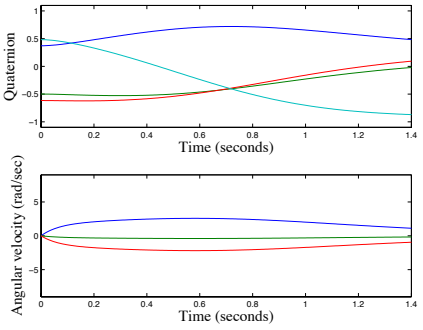


Fig. 4 Response of DMP system (18), (9) with quaternion difference instead of the logarithmic map and without nonlinear term $\mathbf{f}_o$ (Pastor et al, 2011).

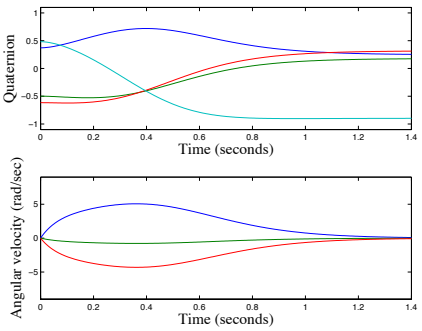


Fig. 5 Response of DMP system (19), (9) with double quaternion difference and without nonlinear term $\mathbf{f}_o$.

Since forces and torques are used only as desired variables along the trajectory and not as robot control variables, they do not need to be encoded by DMPs. Instead we use linear combinations of radial basis functions

$$\mathbf{F}_d(x) = \frac{\sum_i \mathbf{w}_i^F \Psi_i(x)}{\sum_i \Psi_i(x)} x, \tag{24}$$

$$\mathbf{M}_d(x) = \frac{\sum_i \mathbf{w}_i^M \Psi_i(x)}{\sum_i \Psi_i(x)} x, \tag{25}$$

to approximate the desired forces and torques along the phase $x_i = x(t_i)$. Six systems of linear equations need to be solved in order to estimate $\mathbf{F}_d$ and $\mathbf{M}_d$ from the measured force/torque data (2).

3.2 Error Feedback and DMP Phase Stopping

When the robot executes the demonstrated trajectory, the resulting forces and torques differ from the ones that were measured during human demonstration. This occurs due to small displacements arising from inaccurate pose estimation, reduced accuracy of the robot tracking control due to changes in the robot configuration (note that the original trajectory must be moved to a new configuration as specified by the workpiece pose), and uncertainties in the placement of the peg in the gripper. This could worsen or even prevent a successful execution of the PiH task. In order to adapt to a new situation, we propose to modify the demonstrated trajectory according to the admittance (Villani and De Schutter, 2008) PI control law (indirect compliance, also referred to as classical impedance, (Yoshikawa, 2000))

$$\mathbf{p}_c(x) = \mathbf{K}_{s1}\mathbf{e}_p(x) + \mathbf{K}_{s3}\mathbf{ei}_p(x) + \boldsymbol{\varphi}_p(x) + \mathbf{p}_r(x), \tag{26}$$

$$\mathbf{q}_c(x) = \chi(\mathbf{K}_{s2}\mathbf{e}_q(x) + \mathbf{K}_{s4}\mathbf{ei}_q(x)) * \boldsymbol{\varphi}_q(x) * \mathbf{q}_r(x), \tag{27}$$

where $\mathbf{p}_c(x)$ and $\mathbf{q}_c(x)$ are the commanded position and quaternion fed to the robot controller, $(\mathbf{p}_r(x), \mathbf{q}_r(x))$ is the reference trajectory, i. e. the displaced demonstrated trajectory computed by integrating the learned DMPs ($\mathbf{p}_{DMP}$ and $\mathbf{q}_{DMP}$) and then applying the workpiece displacement $(\Delta\mathbf{t}_w, \Delta\mathbf{q}_w)$,

$$\mathbf{p}_r(x) = \Delta\mathbf{q}_w * \mathbf{p}_{DMP}(x) * \overline{\Delta\mathbf{q}}_w + \Delta\mathbf{t}_w,$$
$$\mathbf{q}_r(x) = \Delta\mathbf{q}_w * \mathbf{q}_{DMP}(x).$$

In our system, the workpiece displacement is estimated by an RGB-D camera. $\mathbf{K}_{s1}, \mathbf{K}_{s2}, \mathbf{K}_{s3}, \mathbf{K}_{s4} \in \mathbb{R}^{3\times3}$ are positive definite, diagonal gain matrices. The feedback error terms, i. e. $\mathbf{K}_{s1}\mathbf{e}_p(x)+\mathbf{K}_{s3}\mathbf{ei}_p(x)$ and $\chi(\mathbf{K}_{s2}\mathbf{e}_q(x)+\mathbf{K}_{s4}\mathbf{ei}_q(x))$, respectively provide force/torque feedback control. Vectors $\boldsymbol{\varphi}_p(x)$ and $\boldsymbol{\varphi}_q(x)$ denote additional translational and rotational displacement, respectively, which are learned on line (see Section 3.3). Initially, they are set to $\boldsymbol{\varphi}_p = [0,0,0]^{\mathrm{T}}$ and $\boldsymbol{\varphi}_q = 1 + [0,0,0]^{\mathrm{T}}$. The main idea of our approach is that learning should move as much of the feedback error as possible to these displacement trajectories. The errors $\mathbf{e}_p(x)$, $\mathbf{e}_q(x) \in \mathbb{R}^3$ are defined as

$$\mathbf{e}_p(x) = \mathbf{q}(x) * (\mathbf{F}_d(x) - \mathbf{F}) * \overline{\mathbf{q}(x)}, \tag{28}$$

$$\mathbf{e}_q(x) = \mathbf{q}(x) * (\mathbf{M}_d(x) - \mathbf{M}) * \overline{\mathbf{q}(x)}, \tag{29}$$

where $\mathbf{F}_d(x)$ and $\mathbf{M}_d(x)$ are the desired reference force and torque at phase x as acquired from human demonstration, $\mathbf{F}$ and $\mathbf{M}$ the current measured force and torque, and $\mathbf{q}(x)$ the quaternion specifying the current orientation of the robot's tool. Note that for any unit quaternion $\mathbf{q} \in \mathrm{S}^3$ and $\mathbf{p} \in \mathbb{R}^3$, the expression $\mathbf{q}*\mathbf{p}*\overline{\mathbf{q}}$ results in a vector $\mathbf{p}$ rotated by a rotation described by quaternion $\mathbf{q}$. When combing 3-D vectors and quaternions like in quaternion multiplication of Eq. (28) and (29), 3-D vectors are interpreted as quaternions with the scalar part equal to zero.

The integrated errors $\mathbf{ei}_p(x)$ and $\mathbf{ei}_q(x) \in \mathbb{R}^3$ are cumulative sums of the corresponding errors $\mathbf{e}_p(x)$ and

$\mathbf{e}_q(x)$ from the start of the robot movement to the current phase x_j. They are calculated as

$$\mathbf{ei}_p(x_j) = \sum_{\imath=1}^{j} \mathbf{e}_p(x_i), \tag{30}$$

$$\mathbf{ei}_q(x_j) = \sum_{\imath=1}^{j} \mathbf{e}_q(x_j). \tag{31}$$

The proposed controller tracks and adapts the desired positions and orientations while trying to achieve the same forces and torques as the trained movement. Force/torque adaptation requires low gains (matrices $\mathbf{K}_s$) for stable and robust operation. Thus, force adaptation is usually slow. In order to effectively minimize the force/torque error, we propose slowing down the trajectory execution using DMP slow-down feedback. For DMP phase stopping (Ijspeert et al, 2001), the original equation for phase (5) is replaced with

$$\tau\dot{x} = -\frac{\alpha_x x}{1+\alpha_{px}\varepsilon}, \tag{32}$$

where ε is the trajectory tracking error that can be estimated as $\varepsilon = \|\tilde{\mathbf{p}}-\mathbf{p}\|+\gamma\,\mathrm{d}(\tilde{\mathbf{q}}*\overline{\mathbf{q}})$, where $\tilde{\mathbf{p}}$ and $\tilde{\mathbf{q}}$ are the actual position and orientation of the tool center point, respectively, and $\mathbf{p}$ and $\mathbf{q}$ the corresponding DMP outputs. In our task the most relevant error measure are the discrepancies between the desired and actual forces (28) and torques (29), i. e.

$$\varepsilon = \left\| \left[\mathbf{e}_p^{\mathrm{T}}, \mathbf{e}_q^{\mathrm{T}}\right] \right\|. \tag{33}$$

Note that in the case of large force or torque errors, the error value ε becomes large which in turn makes the phase change $\dot{x}$ small. Thus the phase evolution is stopped until the robot reduces the force/torque error. (Ijspeert et al, 2001) proposed to modify also Eq. (4) to ensure faster error reduction

$$\tau\dot{y} = z + \alpha_{py}(\tilde{y} - y), \tag{34}$$

where $\tilde{y}$ denotes the actual position of the robot and y the DMP calculated position. In the context of Cartesian space DMPs, Eq. (34) becomes different for the positional and orientational part of the trajectory, which are respectively encoded by Eq. (21) and (9), i. e.

$$\tau\dot{\mathbf{p}} = \mathbf{z} + \alpha_{pp}(\tilde{\mathbf{p}} - \mathbf{p}), \tag{35}$$

$$\tau\dot{\mathbf{q}} = \frac{1}{2}\left(\boldsymbol{\eta} + \alpha_{pq} 2\log\left(\tilde{\mathbf{q}} * \overline{\mathbf{q}}\right)\right) * \mathbf{q}, \tag{36}$$

In the context of force feedback control we replace trajectory tracking error with force tracking error

$$\tau\dot{\mathbf{p}} = \mathbf{z} - \alpha_{pp}\mathbf{K}_{s1}\mathbf{e}_p(x), \tag{37}$$

$$\tau\dot{\mathbf{q}} = \frac{1}{2}\left(\boldsymbol{\eta} - \alpha_{pq}\mathbf{K}_{s2}\mathbf{e}_q(x)\right) * \mathbf{q}. \tag{38}$$

During the execution of the learned trajectory, the resulting positions and orientations offsets are sampled depending on the phase variable x, which ensures that the sampling is independent of the trajectory duration. Thus, the trajectory is sampled exactly the same number of times as during training, even when the phase is slowed down during the task execution due to the proposed phase stopping mechanism.

3.3 Offset Learning

The goal of learning is to iteratively modify the positional and orientational part of the demonstrated trajectory so that the transformed trajectory results in similar forces and torques as after the application of the training procedure of Section 2. The offset trajectory is updated after each iteration step using the combined offsets

$$\mathbf{s}^p_{j,l+1} = \boldsymbol{\varphi}_{p,l}(x_j) + \mathbf{K}_{s1}\mathbf{e}_p(x_j) + \mathbf{K}_{s3}\mathbf{ei}_p(x_j), \tag{39}$$

$$\mathbf{s}^q_{j,l+1} = \chi(\mathbf{K}_{s2}\mathbf{e}_q(x_j) + \mathbf{K}_{s4}\mathbf{ei}_q(x_j)) * \boldsymbol{\varphi}_{q,l}(x_j), \tag{40}$$

which are used also in (26) and (27) to compute the reference trajectory. Index l denotes the learning trial. Each component φ_k of the offsets $\boldsymbol{\varphi}_{p,l}$ and $\boldsymbol{\varphi}_{q,l}$ of this newly sampled offset trajectory is represented as a linear combination of M radial basis functions (just like the DMPs in Eq. (6))

$$\varphi_k(x) = \frac{\sum_{i=1}^{M} w_{i,k}\Psi_i(x)}{\sum_{i=1}^{M}\Psi_i(x)}x. \tag{41}$$

The new data points $\{s^k_{j,l+1}\}$, $j = 0,\ldots,T$, are obtained from the k-th component of the offsets trajectory, where $k = 1,2,3$, denote the components of the positional part $\mathbf{s}^p_{j,l+1}$ and $k = 4,5,6,7$, the components of the quaternion part $\mathbf{s}^q_{j,l+1}$. The aim of optimization is to find weights $\{w_{i,k}\}$ that minimize the quadratic cost function

$$\sum_{j=0}^{T}(\varphi_k(x_j) - s^k_{j,l+1})^2. \tag{42}$$

For each index k the optimal weights are computed by solving the following system of linear equations

$$\mathbf{A}\mathbf{w}_k = \mathbf{s}_k, \tag{43}$$

$$\mathbf{w}_k = \begin{bmatrix} w_{1,k} \\ \vdots \\ w_{M,k} \end{bmatrix}, \qquad \mathbf{s}_k = \begin{bmatrix} s^k_{0,l+1} \\ \vdots \\ s^k_{T,l+1} \end{bmatrix},$$

$$\mathbf{A} = \begin{bmatrix} \frac{\psi_1(x_0)x_0}{\sum_{i=1}^{M}\psi_i(x_0)} & \cdots & \frac{\psi_M(x_0)x_0}{\sum_{i=1}^{M}\psi_i(x_0)} \\ \vdots & \ddots & \vdots \\ \frac{\psi_1(x_T)x_T}{\sum_{i=1}^{M}\psi_i(x_T)} & \cdots & \frac{\psi_M(x_T)x_T}{\sum_{i=1}^{M}\psi_i(x_T)} \end{bmatrix}.$$

ψ_i are the Gaussian kernel functions from (7). Note that the quaternion part of the offset trajectory $\boldsymbol{\varphi}_q$ has to be normalized before being used in (27).

3.4 Control Scheme

Fig. 6 shows the control flow diagram for the proposed learning system. In this diagram, $\mathcal{G}_d(t)$ denotes the original demonstrated trajectory, which is encoded by DMPs. $\mathcal{Q}_d(x)$ refers to the output signal obtained by integrating (20), (37), (8), (38), (32) and by applying the workpiece displacement $(\Delta\mathbf{t}_w, \Delta\mathbf{q}_w)$ as estimated by vision, which occurs in block T_1. $\mathcal{F}_d(t)$ denotes the reference force/torque acquired during training and $\mathcal{F}_d(x)$ denotes the output calculated by using Eq. (24) and (25). Due to the noise induced by vision, robot tracking errors, and uncertainties in the grasp configuration, the current forces $\mathbf{F}$ and torques $\mathbf{M}$ measured during the task execution differ from the trained force/torque profiles $\mathcal{F}_d$. The aim of the learning process is to reduce the difference between the measured and trained forces and torques. In T_2 the measured force error is transformed into robot base coordinates. Using admittance PI control (26) – (27), the position/orientation offset is calculated and added to the existing offset from the previous iteration step. This offset is computed using function approximation in the block $\boldsymbol{\varphi}(x) = \{\boldsymbol{\varphi}_p(x), \boldsymbol{\varphi}_q(x)\}$. Motor commands $\mathcal{Q}_c$ are given as the aggregation of the DMP generated trajectory, force feedback (26) – (27) and of the offset learned in several iterations (39) – (40). This process is repeated until the measured forces match the desired forces or no further improvement is possible. The proposed learning algorithm (gray shaded in Fig. 6), belongs to a class of iterative learning control algorithms, where we applied the current iteration causal learning as described by (Bristow et al, 2006; Moore et al, 2006). Another related approach is feedback error learning (Kawato, 1990; Nakanishi and Schaal, 2004).

In our approach, the originally trained trajectory is always preserved and we learn an offset to this trajectory rather than modifying the trajectory itself. In such a setting, the offset can be easily reset when a robot encounters a new situations.

3.5 Trajectory Adaptation Using Impedance Control Law

In the previous section we introduced a learning framework based on the admittance control law. The benefit of this approach is that it can be implemented also on

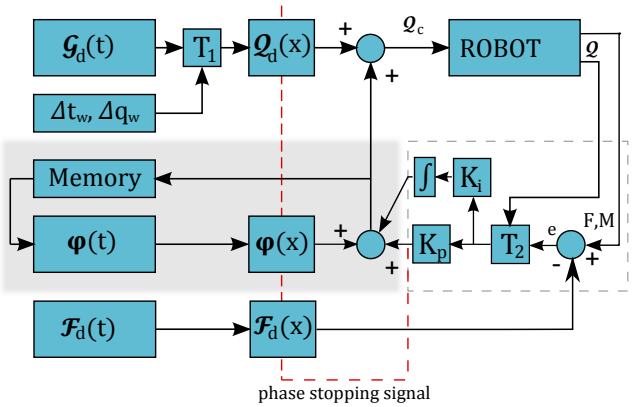


Fig. 6 Flow chart of the complete learning system

velocity controlled robots without access to the torque level control. In this section we extend our learning approach to kinematically redundant robots with n degrees of freedom controlled by the impedance control law, which is preferable if allowed by the robot.

Impedance control equations are derived in Appendix A. By rewriting (51) and (52) we can express the task space error in the form

$$\begin{aligned}-\mathbf{K}_{s1}(\mathbf{q} * (\mathbf{F}_d - \mathbf{F}) * \overline{\mathbf{q}}) &= \ddot{\mathbf{p}}_r - \ddot{\mathbf{p}} + \mathbf{K}_d(\dot{\mathbf{p}}_r - \dot{\mathbf{p}}) + \\ &\quad \mathbf{K}_p(\mathbf{p}_r - \mathbf{p}),\\ -\mathbf{K}_{s2}(\mathbf{q} * (\mathbf{M}_d - \mathbf{M}) * \overline{\mathbf{q}}) &= \dot{\boldsymbol{\omega}}_r - \dot{\boldsymbol{\omega}} + \mathbf{K}_q^d(\boldsymbol{\omega}_r - \boldsymbol{\omega}) + \\ &\quad \mathbf{K}_q^p \log(\mathbf{q}_r * \overline{\mathbf{q}}),\end{aligned}$$

where $\mathbf{q}$ is the current quaternion specifying the tool orientation. In the above equation, subscript $_r$ denotes the reference trajectory values, subscript $_d$ the desired forces and torques, and variables without the subscript the current values as measured from the robot. The above error equations show that it is necessary to assure zero position and orientation error in order to obtain zero force error. This happens if the reference trajectory $\mathbf{p}_r$ and $\mathbf{q}_r$ is identical to the actual robot trajectory $\mathbf{p}$ and $\mathbf{q}$. We thus add positional and orientational offsets to the reference DMP trajectory

$$\mathbf{p}_{r,l}(x) = \boldsymbol{\varphi}_{p,l}(x) + \mathbf{p}_r(x), \tag{44}$$

$$\mathbf{q}_{r,l}(x) = \boldsymbol{\varphi}_{q,l}(x) * \mathbf{q}_r(x). \tag{45}$$

To make the reference trajectory closer to the actual robot trajectory, we estimate the new offsets as follows

$$\mathbf{s}^p_{j,l+1} = (\mathbf{p}_l(x_j) - \mathbf{p}_{l-1}(x_j)) + \boldsymbol{\varphi}_{p,l}(x_j), \tag{46}$$

$$\mathbf{s}^q_{j,l+1} = (\mathbf{q}_l(x_j) * \overline{\mathbf{q}}_{l-1}(x_j)) * \boldsymbol{\varphi}_{q,l}(x_j), \tag{47}$$

where $(\mathbf{p}_l(x),\ \mathbf{q}_l(x))$ is the actual robot trajectory in the l-th learning step. Using these offsets, we update the offset terms $\boldsymbol{\varphi}_p(x)$ and $\boldsymbol{\varphi}_q(x)$ in exactly the same way as in Section 3.3 after each task execution. The only difference between the two approaches is that here the feedback control that generates data for offset learning is based on impedance instead of admittance control.

4 Experimental evaluation

The proposed paradigm for learning and adpatation of force-based skills was implemented on two different robot platforms, shown in Fig. 1 and 2, respectively:

- A 6 DOF Universal Robot arm - type UR5 - equipped with a SCHUNK SDH gripper and a force/torque wrist sensor. This robot uses a high gain non compliant controller. Therefore, we implemented admittance control law as defined by Eq. (26) and (27).
- A 7 DOF Kuka lightweight robot arm equipped with a two-finger jaw gripper. This robot can measure joint torques and can therefore be controlled by Cartesian compliance control law, as described in Section 3.5.

4.1 Evaluation of the applied control schemes

In order to demonstrate the effectiveness of learning in conjunction with force control, we first evaluated the proposed algorithms on a simple task, where the robot had to track sinusoidal force in z direction while standing still. The initial contact force was 10 N. The experiments were conducted with the KUKA lightweight robot arm. The robot's stiffness was set to 3000 N/m. We evaluated the quality of force tracking with the following control laws applied: 1) admittance P-control law, i. e. PI control law with terms $\mathbf{K}_{s3}$ and $\mathbf{K}_{s4}$ set to zero, 2) admittance PI-control law (both of them defined through Eq. (26) – (27)), and 3) impedance control law (defined by Eq. (51) – (52)) without learning. As we can see from Fig. 7, best tracking was achieved with the PI control law. The integral term cancels steady state error, but on the other hand introduces additional phase lag. This is a severe problem when the robot needs to track unknown surface shape with force control, since phase lag can result in significantly increased contact forces. Please note also that the integral term gain $\mathbf{K}_{s3}$ for this experiment was selected at the stability margin and any additional increase in this gain would results in closed loop instability.

We repeated the same experiment with the proposed iterative learning. To ensure stable learning, the integral gain $\mathbf{K}_{s3}$ had to be divided by a factor of 5. Results for all three control laws in conjunction with learning are show in Fig. 8. In all cases, learning results in almost perfect tracking after 5 learning cycles. As expected, impedance control had the smoothest transient response, which can be observed in the first 0.05 sec of the experiment. PI control law still exhibits small phase lag, which indicates that the integral gain $\mathbf{K}_{s3}$ should be kept low for stable operation. On the other hand, integral term is particulary effective in combination with phase stopping, since it can successfully eliminate the steady state error.

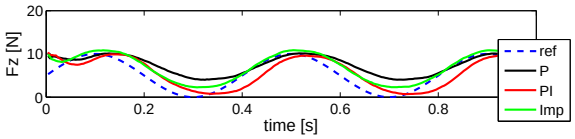


Fig. 7 Comparison of stiffness P, stiffness PI and impedance control in tracking sinusoidal force profile

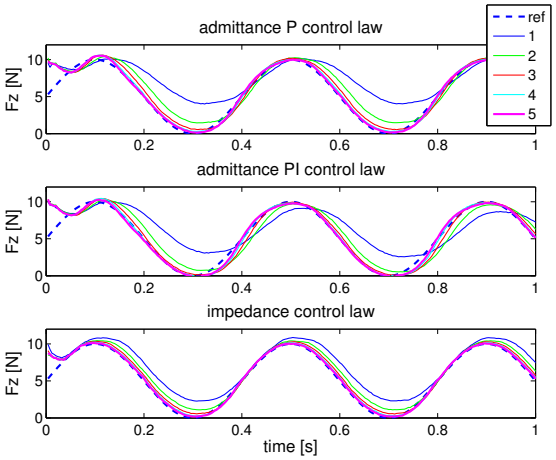


Fig. 8 Tracking of sinusoidal force profile 5 consecutive learning cycles for stiffness P, stiffness PI and impedance control

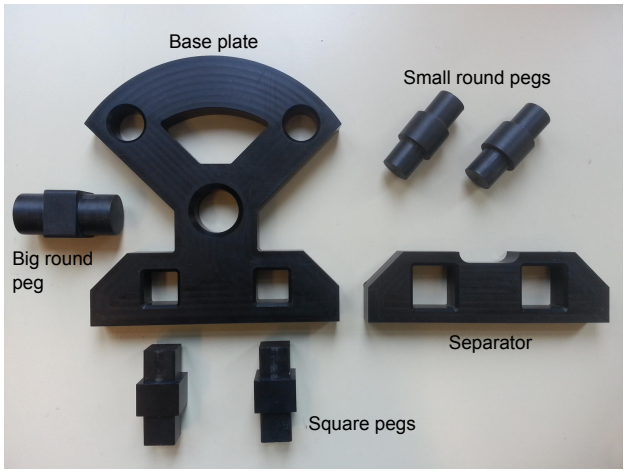
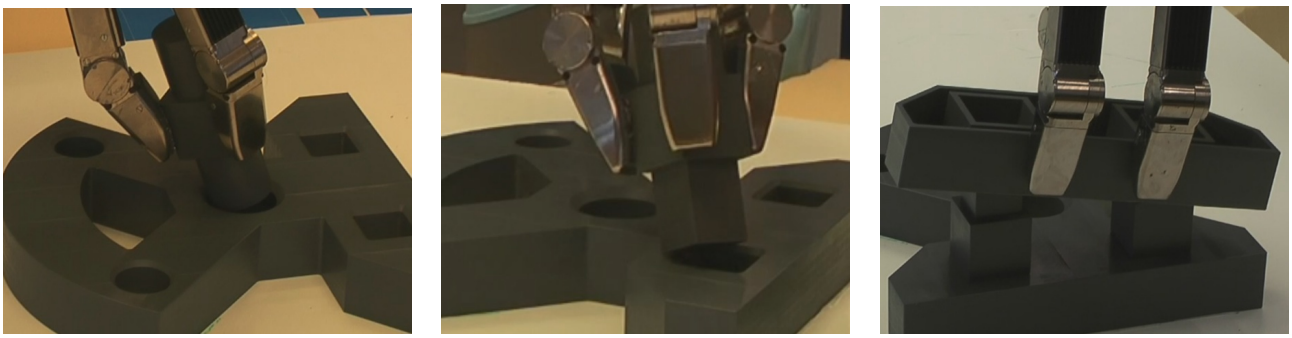


Fig. 10 Cranfield assembly benchmark.

4.2 Evaluation on Peg-in-Hole Task

The tasks for evaluation were taken from the Cranfield assembly benchmark (Collins et al, 1985) and included the insertion of different pegs (round and square) into the corresponding holes of the base plate and the placement of a separator onto two previously inserted pegs (see Fig. 9 and 10). Initial trajectories and force/torque profiles were obtained using tele-operation and kinesthetic guiding as described in Section 2. We executed



(a) Round peg. (b) Square peg. (c) Separator.

Fig. 9 The insertion and placement of different piece shapes with the Universal Robot arm.

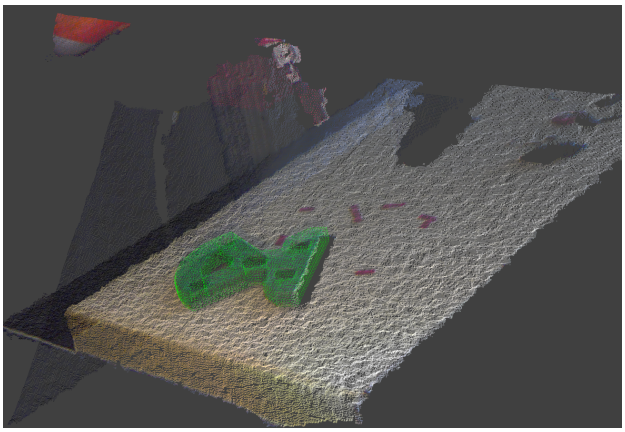


Fig. 11 Kinect/PCL - based visual pose estimation.

the trajectories for different randomly selected positions of the base plate, which were estimated by a 3-D vision system. The training data were translated and rotated into new configurations using the pose of the base plate as estimated by vision (Fig. 11). In this scenario most of the uncertainties come from vision. Therefore, the resulting forces during the execution of the demonstrated trajectory may be different from the ones measured during training.

Fig. 11 illustrates the localization of the base plate within the workspace by a 3-D visual system using off the shelf range sensor. A CAD model of the base plate is provided to the system and fitted within the 3-D scene, thus determining both the position and orientation of the object in the world coordinate frame. The robot uses this information to target particular holes in the base plate for peg insertion. Object localization is performed using a component of the Point Cloud Library (PCL) (Rusu and Cousins, 2011) for point cloud manipulation.

4.2.1 Peg-in-Hole with Admittance Control

The learning and adaptation procedure on the Universal robot arm UR5, which allows only admittance control, was implemented in C++ and linked to the robot using ROS (Quigley et al, 2009). After training we slightly displaced the base plate and used the acquired training data to execute the task. The results for square peg insertion of Fig. 9(b) are depicted in Fig. 12 – 14. Fig. 12 shows the learned offsets of the positional part of the trajectory in five consecutive cycles. Fig. 13 shows the forces that result from the execution of the original (demonstrated) trajectories and the adjusted (learned) trajectories. Although force-based feedback control was used in all task executions, the adjusted trajectories after a few cycles generally result in significantly lower forces and torques, as evident from Fig. 13. Fig. 14 illustrates the phase evolution in consecutive learning cycles. The original phase measured during human demonstration is shown as dotted line. Whenever the difference between the measured and the desired forces became large enough, the phase was slowed down due to the force/torque error term in Eq. (32) until the force feedback controller sufficiently reduced it. The resulting offsets were learned as described in Section 3.3. Since our learning procedure reduces the difference between the desired and measured forces and torques, the phase stopping becomes less frequent and consequently the execution time decreases in each learning cycle, as can be seen in Fig. 14. Even though the contact forces were highly elevated in the first trial (Fig. 13), the proposed approach succeeded to significantly reduce force errors and consequently reduce the insertion time. This indicates the robustness of our algorithm.

In order to demonstrate that the developed approach does not depend on the shape of the workpiece, we applied it also to PiH tasks involving other types of pegs, e. g. round peg of Fig. 9(a). The results shown in Fig. 15 – 17 demonstrate that the developed sys-

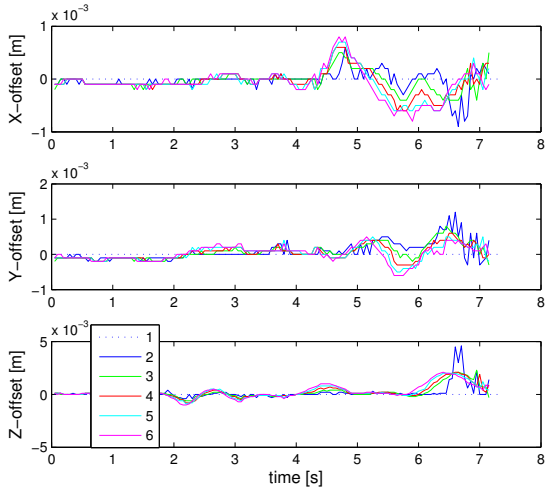


Fig. 12 Positional part of the learned offset in 5 consecutive cycles. Cycle 1 denotes the execution of the trained trajectory (zero offset).

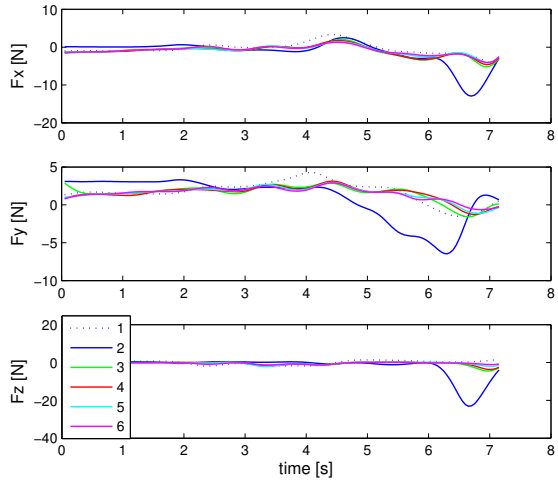


Fig. 13 Forces measured during the square PiH task. Dashed lines (cycle 1) correspond to the original DMP trajectory and solid lines to the adjusted trajectories in 5 consecutive learning cycles.

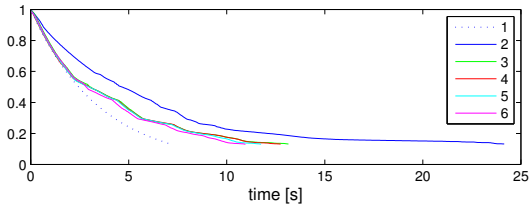


Fig. 14 Phase evolution during the square PiH task. Dashed line is the phase evolution achieved after training and solid lines are the actual phase in 5 consecutive cycles.

tem works equally well for round pegs and is therefore suitable to specify insertion strategies for workpieces of different shapes. Although different geometries require different peg insertion strategies, the proposed training and learning procedure can be used for all of them.

Yet a more complex workpiece with two holes is a separator, which needs to be put on two square pegs priorly inserted into the base plate (see Fig. 9(c) and 10). Even though this is a significantly more complex task, the same learning and adaptation procedure could be applied to perform the task, thus exhibiting the generality of the proposed approach. The results shown in Fig. 18 – 20 are comparable to the results obtained during square and round peg insertion. Note that since unexpectedly large forces (F_y component in Fig. 19) were exerted at the end of the separator placement in the first trial, the separator slipped in the robot hand. Consequently, additional forces arose in the next attempt, but these were successfully compensated for in the later cycles. This demonstrates that our approach can deal also with unforeseen perturbations that cause deviations from the original force profile, such as the slippage of the object in the robot hand described in the above experiment.

4.2.2 Peg-in-Hole with Impedance Control

Additional experiments were conducted on Kuka LWR arm, which unlike Universal robot arm enables impedance control. The learning procedure was implemented in Matlab, which communicated with the Kuka LWR controller using Fast Research Interface (Schreiber et al, 2010). Fig. 21 – 23 show the experimental results of the algorithm performing round PiH with the Kuka robot. The obtained results are comparable with the results obtained with the Universal robot, but the insertion is generally smoother and more robust due to the Cartesian impedance control. The ability to execute the desired behaviors faster is an important advantage of impedance control.

Square peg insertion has also been tested with Kuka robot (Fig. 24–26). By observing the phase evolution in Fig. 26, we can see that the algorithm is gradually decreasing the execution time, which becomes quite comparable to the execution time after training.

4.2.3 Statistical Evaluation

Further experiments were carried out with Kuka LWR robot arm to statistically evaluate the algorithm's efficiency. PiH execution was tested using 120 experiments on 8 recorded trajectories with both square and round pegs. The base plate was randomly put at different locations in the area of 0.25 m^2 on the table within the robot's workspace. The poses of the base plate were estimated by vision. The statistics of results for square

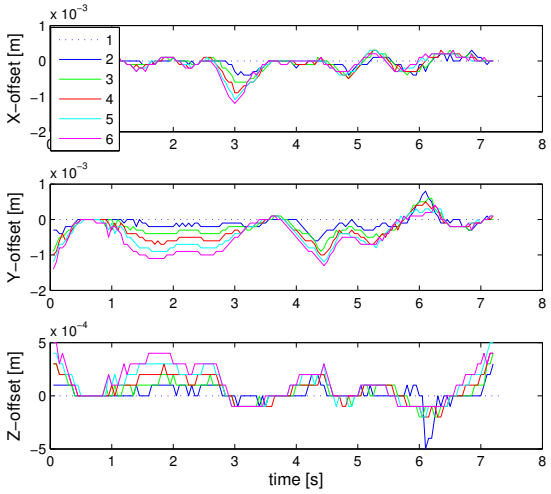


Fig. 15 Positional part of the learned offset in 5 consecutive cycles. Cycle 1 denotes zero offset execution.

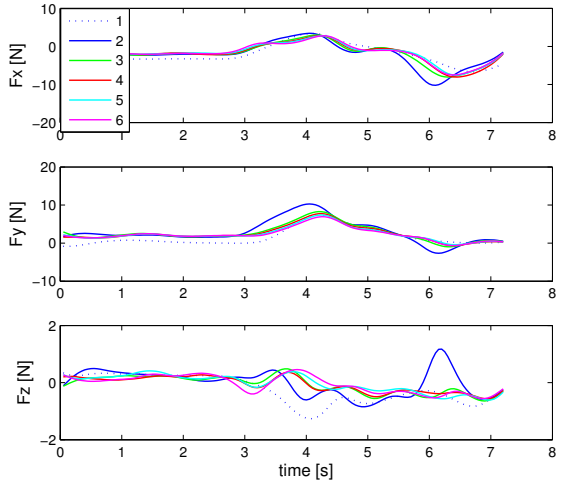


Fig. 16 Forces measured during the round PiH task in 5 consecutive learning cycles.

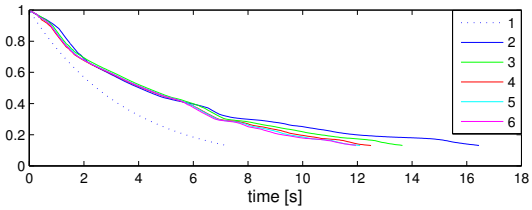


Fig. 17 Phase evolution during the round PiH task in 5 consecutive cycles.

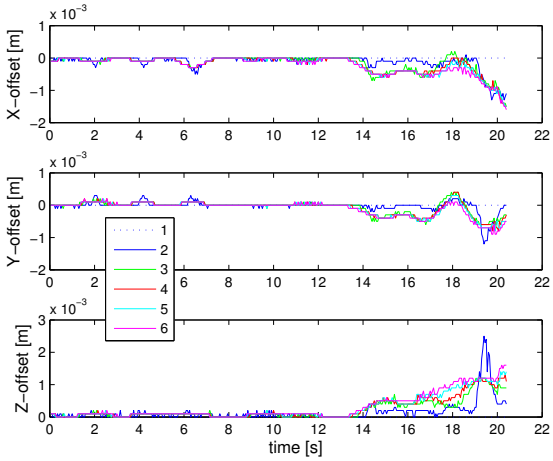


Fig. 18 Positional part of the learned offset in 5 consecutive cycles. Cycle 1 denotes zero offset execution.

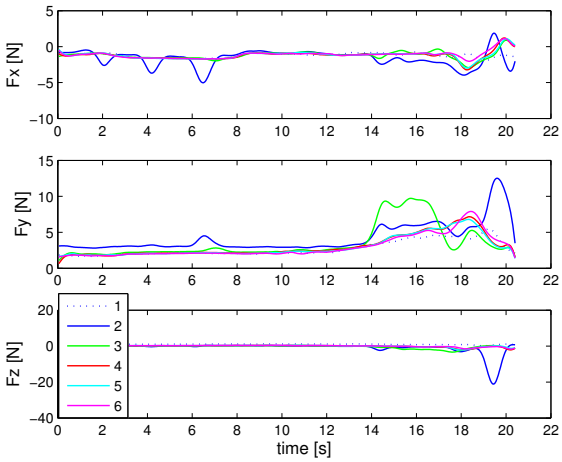


Fig. 19 Forces measured during the separator placement in 5 consecutive learning cycles.

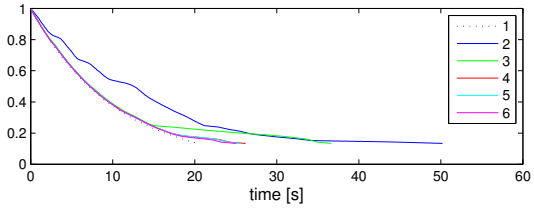


Fig. 20 Phase evolution during the separator placement in 5 consecutive cycles. Since in this experiment the base plate was moved to a new location and its location was estimated by vision, the initial performance is much slower in this case.

and round pegs are shown in Fig. 27 and 29, respectively. We can observe that most of adaptation is performed in the first three cycles, which indicates the fast convergence of the algorithm.

The statistical summary of these experiments shows that the force discrepancies are reduced by more than 56% percentage on average over 5 trials. Moreover, by analyzing the execution time in Fig. 28 and 30 we can see that it drops by more than 30% after the first trial, where most of the adaptation has occurred. However, the execution time can still be significantly improved in subsequent adaptation steps. Final reduction of the execution time reaches up to 50% over 5 trials.

50 additional experiments were performed with Kuka LWR robot arm to compare the success rate of the algorithm with phase stopping and without it. PiH exe-

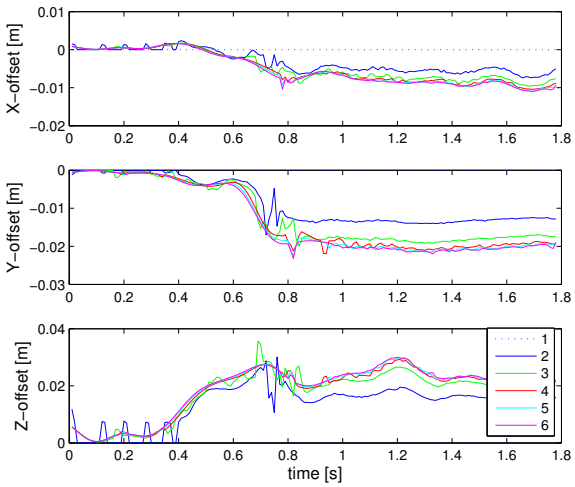


Fig. 21 Positional part of the learned offset in 5 consecutive cycles. Cycle 1 denotes the execution of the trained trajectory (zero offset).

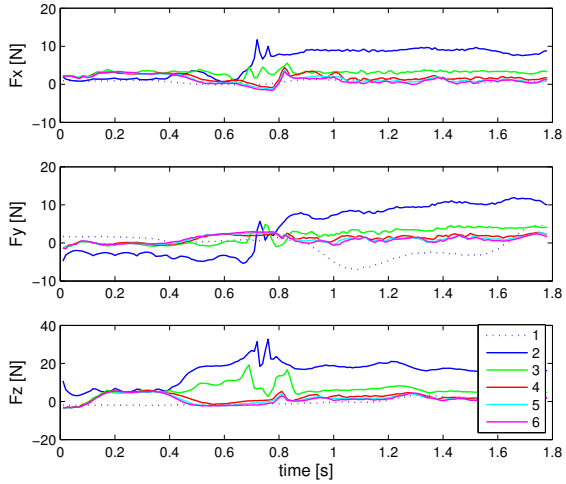


Fig. 22 Forces measured during the execution of the square PiH task. Dashed lines (cycle 1) correspond to the original DMP trajectory and solid lines to the adjusted trajectories in 5 consecutive learning cycles.

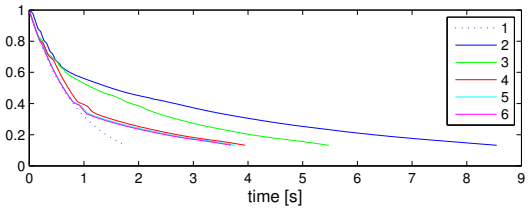


Fig. 23 Phase evolution during the execution of the square PiH task. Dashed line corresponds to the ideal phase evolution achieved after training and solid lines to the actual phase in 5 consecutive cycles.

cution was tested on 5 recorded trajectories with round pegs. The base plate was randomly put at different locations in the area of 0.25 m^2 on the table within the robot's workspace. The poses of the base plate were es-

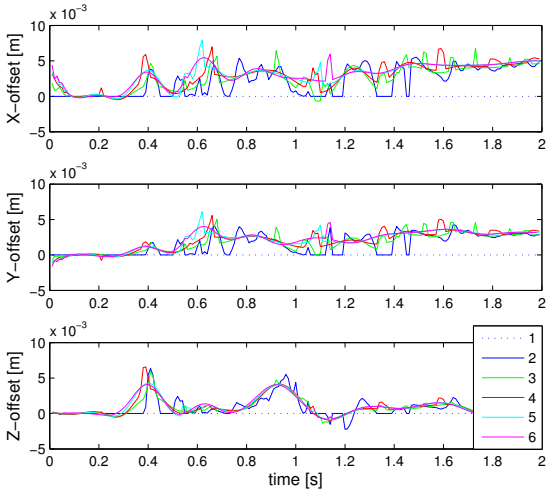


Fig. 24 Positional part of the learned offset in 5 consecutive cycles for square PiH.

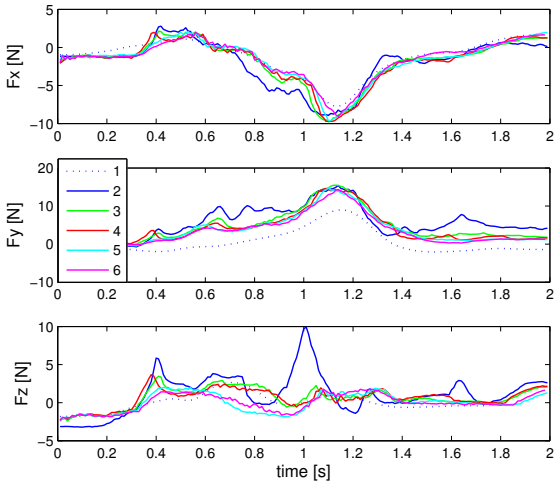


Fig. 25 Forces measured during the execution of the square PiH task in 5 consecutive learning cycles.

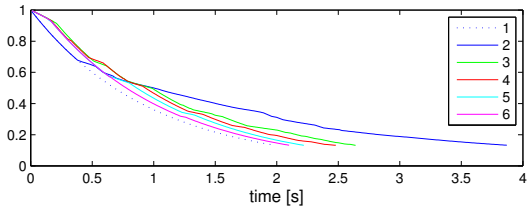


Fig. 26 Phase evolution during the execution of the square PiH task in 5 consecutive cycles.

timated by vision. The comparison of the success rate with and without phase stopping are shown in Fig. 31. It can be clearly observed that the application of phase stoping improves the reliability of the approach. Fig. 32 depicts how the robots follows one of the trained trajectories with and without phase stopping. Successful tracking in z-coordinate, which is parallel to the hole's axis, indicates successful insertion. The deviations in x

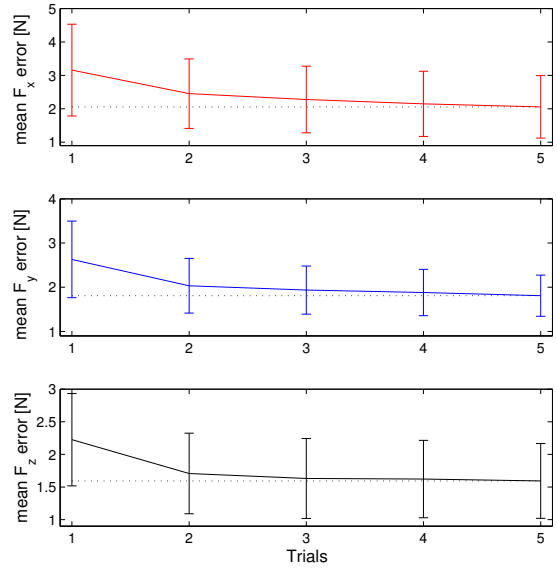


Fig. 27 Mean force errors of the square peg insertion.

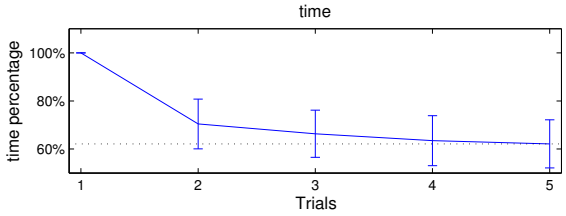


Fig. 28 Percentage of the average time needed for the square peg insertion.

and y coordinates show the corrections made to achieve successful insertion. In this case peg insertion was not achieved without phase stopping, as demonstrated by much shallower value of z. Failures in the cases with phase stopping were all due to the peg totally missing the hole. In this case force sensing does not provide any information about how to move to achieve successful insertion. 100 % success rate could be achieved in this experiment with the additional search algorithms briefly described in (Savarimuthu et al, 2013), but this is beyond the scope of this paper.

5 Conclusion

We proposed a new approach for learning of force-based manipulation skills. The main feature of our approach is that it takes into account Cartesian space trajectories *and* force/torque profiles arising during the execution of the task. The main contributions are:

– An algorithm that uses force feedback and DMP phase stopping to modify the trained trajectories within the robot's servo loop so that the resulting forces and torques become more similar to the forces and torques obtained during training. The DMP phase stopping mechanism is essential to enable smooth and reliable adaptation. It gives to the integral term in the force adaptation loop enough time to diminish the tracking error.
– An iterative adaptation algorithm based on the current iteration causal learning that gradually transfers the force/torque feedback error term to the off-

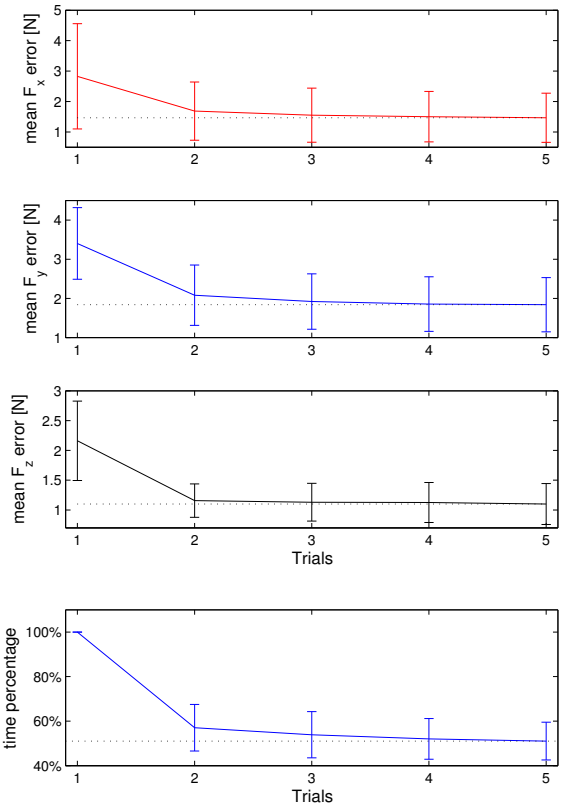


Fig. 30 Percentage of the average time needed for the round peg insertion.

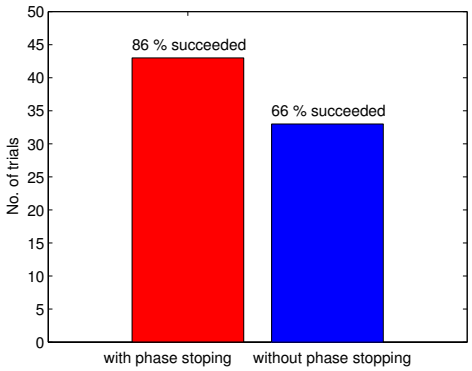


Fig. 31 Success rate comparison between the cases of PiH execution with/without phase stopping.

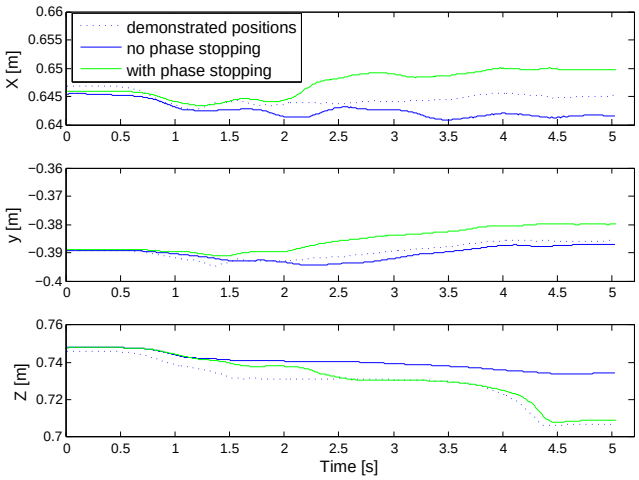


Fig. 32 Comparison of position trajectories originating from a successful insertion with phase stopping and a failed one without phase stopping.

set trajectory, thus minimizing the difference between the measured and the trained forces and torques. If noise is introduced into the system, e. g. by vision, the proposed learning algorithm initially slows down the execution to ensure success, but after learning the speed of execution can again be significantly increased.
- Modification of DMP differential equations in unit quaternion space and the introduction of DMP phase stopping for quaternions based on the logarithmic map.
- Experimental evaluation of the proposed approach including the use of 3-D vision and workpieces with different geometry and different number of holes.

Our experimental results show that the developed system can effectively learn complex manipulation tasks that involve contact with the environment. New manipulation behaviors can quickly be learned by human demonstration and later be adapted to different configurations of the workspace. The flexibility of the approach is demonstrated by training a variety of assembly tasks involving different workpiece geometries and different number of holes. Finally, we have shown that the proposed approach is suitable both for robots that allow impedance control and for robots where only admittance control is possible.

Especially in industry, there are a lot of operations that need to be executed many times in exactly the same configuration. Our algorithm provides a methodology that firstly, enables the robot to quickly acquire new force-based skills by human demonstration, and secondly, to apply the new skill in the robot's workspace using 3-D vision. In this way we provide tools to drastically reduce setup times of robotic workcells in industry. Even in natural environments some tasks need to be repeated. Our approach ensures that the task will be successfully executed already in the first attempt, but slower and sub-optimal. In the next attempt (not necessarily next time, the robot might do other work in the meantime) the robot will improve the policy associated with this position (orientation) and will therefore learn the optimal policies for any (many) position of the workpiece. Generalization from multiple demonstrations is the subject of our future work.

A related approach was presented by Pastor et al. (Pastor et al, 2011), where the interaction with the environment is implemented as modulation of DMPs at the acceleration level, allowing for reactive and compliant behaviors. Unlike in our approach, feedback response to the environment changes was learned only from user demonstration and no additional parameters were introduced into the system. Hence iterative learning to improve the initial response to the environment changes could not be implemented in this approach.

Besides adapting the demonstrated trajectories, it is also possible to refine the demonstrated force/torque profiles and learn the optimal stiffness parameters. The initial force/torque profiles can be refined by means of reinforcement learning as suggested by (Kalakrishnan et al, 2011). Optimal stiffness parameters can also be obtained by reinforcement learning (Buchli et al, 2011). These kind of approaches can be used to improve the effectiveness of the proposed algorithm.

Acknowledgements The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement No. 269959, IntellAct, and No. 600578, ACAT.

A Impedance control

In general, the dynamics of a robot arm interacting with the environment is described by

$$\boldsymbol{\rho} = \mathbf{H}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \mathbf{h}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + \mathbf{J}(\boldsymbol{\theta})^{\mathrm{T}} \left[\mathbf{F}^{\mathrm{T}}, \mathbf{M}^{\mathrm{T}}\right]^{\mathrm{T}} \tag{48}$$

where $\boldsymbol{\rho}$ is a vector of joint torques, $\mathbf{H} \in \mathbb{R}^{n \times n}$ is a symmetric, positive definite inertia matrix, $\mathbf{h} \in \mathbb{R}^n$ contains nonlinear terms due to the centrifugal, Coriolis, friction and gravity forces, $\mathbf{J} \in \mathbb{R}^{6 \times n}$ is the robot Jacobian, and $\mathbf{F}$, $\mathbf{M} \in \mathbb{R}^3$ are the vectors of environment contact forces and torques acting on the robot's end-effector. $\boldsymbol{\theta} \in \mathbb{R}^n$ denotes the joint angles. The following relationship holds between joint space and Cartesian space accelerations (Hsu et al, 1989; Nakanishi et al, 2008)

$$\ddot{\boldsymbol{\theta}} = \mathbf{J}_{\mathbf{H}}^{+} \left(\begin{bmatrix} \ddot{\mathbf{p}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} - \dot{\mathbf{J}}\dot{\boldsymbol{\theta}} \right) + \mathbf{N}\boldsymbol{\xi}, \tag{49}$$

where $\mathbf{J}_{\mathbf{H}}^{+} = \mathbf{H}^{-1}\mathbf{J}^{\mathrm{T}}(\mathbf{J}\mathbf{H}^{-1}\mathbf{J}^{\mathrm{T}})^{+} = \mathbf{H}^{-1/2}(\mathbf{J}\mathbf{H}^{-1/2})^{+}$ denotes the inertia weighted pseudo-inverse of $\mathbf{J}$ (Nakamura, 1991; Whitney, 1969). $\boldsymbol{\xi}$ is any vector from $\mathbb{R}^n$ and defines the null space motion. By inserting (49) into (48) we obtain a general form control law for a redundant robot (Nemec et al, 2007)

$$\boldsymbol{\rho} = \mathbf{H}\mathbf{J}_{\mathbf{H}}^{+}\left(\begin{bmatrix}\ddot{\mathbf{p}}\\ \dot{\boldsymbol{\omega}}\end{bmatrix} - \dot{\mathbf{J}}\dot{\boldsymbol{\theta}}\right) + \mathbf{H}\mathbf{N}\boldsymbol{\xi} + \mathbf{h} + \mathbf{J}^{\mathrm{T}}\begin{bmatrix}\mathbf{F}\\ \mathbf{M}\end{bmatrix}, \quad (50)$$

where $\mathbf{N} = (\mathbf{I} - \mathbf{J}_{\mathbf{H}}^{+}\mathbf{J})\mathbf{H}^{-1/2}$ is the projection matrix onto the null space of inertia weighted Jacobian (Nakamura, 1991), i. e. $\mathbf{J}\mathbf{N} = 0$. The first term in Eq. (50) is the task controller, the second term is the null space controller and the third and the fourth term compensate for the non-linear robot dynamics and external forces, respectively. The parameters $\ddot{\mathbf{p}}$, $\dot{\boldsymbol{\omega}}$, and $\boldsymbol{\xi}$ can be used as control inputs that should be set so that the task space tracking errors are minimized. Lets choose the task command inputs $\ddot{\mathbf{p}}_c$, $\dot{\boldsymbol{\omega}}_c$ as follows

$$\ddot{\mathbf{p}}_c = \ddot{\mathbf{p}}_r + \mathbf{K}_p^d(\dot{\mathbf{p}}_r - \dot{\mathbf{p}}) + \mathbf{K}_p^p(\mathbf{p}_r - \mathbf{p}) + \mathbf{K}_{s1}\mathbf{e}_p, \quad (51)$$

$$\dot{\boldsymbol{\omega}}_c = \dot{\boldsymbol{\omega}}_r + \mathbf{K}_q^d(\boldsymbol{\omega}_r - \boldsymbol{\omega}) + \mathbf{K}_q^p \log(\mathbf{q}_r * \overline{\mathbf{q}}) + \mathbf{K}_{s2}\mathbf{e}_q, \quad (52)$$

where subscript $_r$ denotes the reference values and variables without the substrict the current values as received from the robot. The force/torque errors $\mathbf{e}_p$ and $\mathbf{e}_q$ are calculated using (28) and (29), respectively. $\mathbf{K}_p^d$, $\mathbf{K}_q^d$, $\mathbf{K}_p^p$, $\mathbf{K}_q^p$, $\mathbf{K}_{s1}$ and $\mathbf{K}_{s2}$ are positive definite, diagonal positional and rotational damping matrices, positional and rotational stiffness matrices, and force and torque feedback matrices, respectively. The reference position and orientation are calculated by integrating (20), (37), (8), (38), and (32), followed by displacement (28) – (28). The reference velocities and accelerations are calculated as $\dot{\mathbf{p}}_r = \Delta\mathbf{q}_w * \dot{\mathbf{p}}_{DMP} * \overline{\Delta\mathbf{q}}_w$ and $\ddot{\mathbf{p}}_r = \Delta\mathbf{q}_w * \ddot{\mathbf{p}}_{DMP} * \overline{\Delta\mathbf{q}}_w$, respectively, while the reference angular velocities and accelerations are calculated as $\boldsymbol{\omega}_r = \Delta\mathbf{q}_w * \boldsymbol{\eta}_{DMP}/\tau * \overline{\Delta\mathbf{q}}_w$ and $\dot{\boldsymbol{\omega}}_r = \Delta\mathbf{q}_w * \dot{\boldsymbol{\eta}}_{DMP}/\tau * \overline{\Delta\mathbf{q}}_w$, respectively.

With the above choice of command accelerations we obtain the well known impedance control law (Hogan, 1985). By choosing the null space command input $\boldsymbol{\xi}_c$, one can control the null space motion of the robot. Simply setting $\boldsymbol{\xi}_c$ to 0 results in non-conservative motion, i. e. the robot will constantly move in null space minimizing the kinetic energy. To prevent the unnecessary joint space motion, the desired velocities should be set to zero, which results in an energy dissipation controller (Khatib, 1987). The command null space motion vector is thus calculated as

$$\boldsymbol{\xi}_c = -\mathbf{K}_N\dot{\boldsymbol{\theta}}, \quad (53)$$

where $\mathbf{K}_N$ is the joint space damping matrix. The commanded torque $\boldsymbol{\rho}_c$ is finally calculated by inserting $\ddot{\boldsymbol{p}}_c$, $\dot{\boldsymbol{\omega}}_c$, $\boldsymbol{\xi}_c$ into (50). All other variables in (50), i. e. $\mathbf{H}$, $\mathbf{J}$, $\mathbf{J}_{\mathbf{H}}$, $\dot{\mathbf{J}}$, $\mathbf{h}$, and $\mathbf{N}$, are calculated at the current values of $\boldsymbol{\theta}$ and $\dot{\boldsymbol{\theta}}$.

References

Bristow D, Tharayil M, Alleyne A (2006) A survey of iterative learning control. IEEE Control Systems Magazine 26(3):96–114

Broenink JF, Tiernego MLJ (1996) Peg-in-Hole assembly using impedance control with a 6 DOF robot. In: Proceedings of the 8th European Simulation Symposium, pp 504–508

Bruyninckx H, Dutre S, De Schutter J (1995) Peg-on-hole: a model based solution to peg and hole alignment. In: IEEE International Conference on Robotics and Automation (ICRA), Nagoya, Japan, vol 2, pp 1919–1924

Buchli J, Stulp F, Theodorou E, Schaal S (2011) Learning variable impedance control. International Journal of Robotics Research 30(7):820–833

Calinon S, Evrard P, Gribovskaya E, Billard A, Kheddar A (2009) Learning collaborative manipulation tasks by demonstration using a haptic interface. In: IEEE International Conference on Advanced Robotics (ICAR), Munich, Germany

Collins K, Palmer AJ, Rathmill K (1985) The development of a European benchmark for the comparison of assembly robot programming systems. In: Rathmill K, MacConail P, O'leary S, Browne J (eds) Robot Technology and Applications, Springer-Verlag, New York, pp 187–199

Dillmann R (2004) Teaching and learning of robot tasks via observation of human performance. Robotics and Autonomous Systems 47(2-3):109–116

Giordano PR, Stemmer A, Arbter K, Albu-Schäffer A (2008) Robotic assembly of complex planar parts: An experimental evaluation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Nice, France, pp 3775–3782

Gullapalli V, Grupen RA, Barto AG (1992) Learning reactive admittance control. In: IEEE International Conference on Robotics and Automation (ICRA), Nice, France, pp 1475–1480

Hamner B, Koterba S, Shi J, Simmons R, Singh S (2010) An autonomous mobile manipulator for assembly tasks. Autonomous Robots 28:131–149

Hersch M, Guenter F, Calinon S, Billard A (2008) Dynamical system modulation for robot learning via kinesthetic demonstrations. IEEE Transactions on Robotics 24(6):1463–1467

Hirana K, Suzuki T, Okuma S (2002) Optimal motion planning for assembly skill based on mixed logical dynamical system. In: 7th International Workshop on Advanced Motion Control, Maribor, Slovenia, pp 359–364

Hogan N (1985) Impedance control: An approach to manipulation: Part I - theory. Journal of dynamic systems, measurement, and control 107(1):1–7

Hsu P, Hauser J, Sastry S (1989) Dynamic control of redundant manipulators. Journal of Robotic Systems 6(2):133–148

Hutter M, Hoepflinger MA, Gehring C, Bloesch M, Remy CD, Siegwart R (2012) Hybrid operational space control for compliant legged systems. In: Robotics: Science and Systems (RSS), Sydney, Australia

Hyon SH, Hale JG, Cheng G (2007) Full-body compliant human-humanoid interaction: Balancing in the presence of unknown external forces. IEEE Transactions on Robotics 23(5):884–898

Ijspeert AJ, Nakanishi J, Schaal S (2001) Nonlinear dynamical systems for imitation with humanoid robots. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids), Tokyo, Japan, pp 219–226

Ijspeert AJ, Nakanishi J, Hoffmann H, Pastor P, Schaal S (2013) Dynamical movement primitives: Learning attractor models for motor behaviors. Neural Computations 25(2):328–373

Kaiser M, Dillmann R (1996) Building elementary robot skills from human demonstration. In: IEEE International Conference on Robotics and Automation (ICRA), Minneapolis, MN, pp 2700–2705

Kalakrishnan M, Righetti L, Pastor P, Schaal S (2011) Learning force control policies for compliant manipulation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), San Francisco, CA, USA, pp 4639–4644

Kawato M (1990) Feedback-error-learning neural network for supervised motor learning. In: Eckmiller E (ed) Advanced neural computers, Elsevier, Amsterdam, pp 365–372

Kazemi M, Valois JS, Bagnell JA, Pollard N (2014) Human-inspired force compliant grasping primitives. Autonomous Robots 37:209–225

Khatib O (1987) A unified approach for motion and force control of robot manipulators: The operational space formulation. IEEE Journal of Robotics and Automation RA-3(1):43–53

Kormushev P, Calinon S, Caldwell DG (2011) Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input. Advanced Robotics 25(5):581–603

Laurin-Kovitz KF, Colgate JE, Carnes SDR (1991) Design of components for programmable passive impedance. In: IEEE International Conference on Robotics and Automation (ICRA), Sacramento, CA, pp 1476–1481

Lee D, Ott C (2011) Incremental kinesthetic teaching of motion primitives using the motion refinement tube. Autonomous Robots 31(2):115–131

Li Y (1997) Hybrid control approach to the peg-in-hole problem. IEEE Robotics and Automation Magazine 4(2):52–60

Lopes A, Almeida F (2008) A force-impedance controlled industrial robot using an active robotic auxiliary device. Robotics and Computer-Integrated Manufacturing 24:299–309

Moore K, Chen Y, Ahn HS (2006) Iterative learning control: A tutorial and big picture view. In: 45th IEEE Conference on Decision and Control, San Diego, CA, USA, pp 2352–2357

Nakamura Y (1991) Advanced Robotics: Redundancy and Optimization. Addison-Wesley, Boston, MA

Nakanishi J, Schaal S (2004) Feedback error learning and nonlinear adaptive control. Neural Networks 17:1453–1465

Nakanishi J, Cory R, Mistry M, Peters J, Schaal S (2008) Operational space control: A theoretical and empirical comparison. The International Journal of Robotics Research 27:737–757

Nemec B, Ude A (2012) Action sequencing using dynamic movement primitives. Robotica 30:837–846

Nemec B, Žlajpah L, Omrčen D (2007) Comparison of null-space and minimal null-space control algorithms. Robotica 25(5):511–520

Newman WS, Branicky MS, Podgurski HA, Chhatpar S, Huang L, Swaminathan J, Zhang H (1999) Force-responsive robotic assembly of transmission components. In: IEEE International Conference on Robotics and Automation (ICRA), Detroit, Michigan, vol 3, pp 2096–2102

Pastor P, Righetti L, Kalakrishnan M, Schaal S (2011) Online movement adaptation based on previous sensor experiences. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), San Francisco, California, pp 365–371

Quigley M, Gerkey B, Conley K, Faust J, Foote T, Leibsz J, Bergery E, Wheeler R, Ng A (2009) **ROS:** an open-source robot operating system. In: ICRA Workshop on Open Source Software, Kobe, Japan

Raibert MH, Craig JJ (1981) Hybrid position/force control of manipulators. Journal of Dynamic Systems, Measurement, and Control 103(2):126–133

Rozo L, Jiménez P, Torras C (2013) A robot learning from demonstration framework to perform force-based manipulation tasks. Inteligent Service Robotics 6:33–51

Rusu RB, Cousins S (2011) 3D is here: Point Cloud Library (PCL). In: IEEE International Conference on Robotics and Automation (ICRA), ICRA Communications, Shanghai, China

Savarimuthu TR, Liljekrans D, Ellekilde LP, Ude A, Nemec B, Krüger N (2013) Analysis of human peg-in-hole executions in a robotic embodiment using uncertain grasps. In: 9th International Workshop on Robot Motion and Control, Wasowo, Poland, pp 233 –239

Schreiber G, Stemmer A, Bischoff R (2010) The fast research interface for the KUKA lightweight robot. In: ICRA Workshop on Innovative Robot Control Architectures for Demanding (Research) Applications – How to Modify and Enhance Commercial Controllers, Anchorage, Alaska

Skubic M, Volz RA (1998) Learning force-based assembly skills from human demonstration for execution in unstructured environments. In: IEEE International Conference on Robotics and Automation (ICRA), Leuven, Belgium, pp 1281–1288

Stemmer A, Albu-Schäffer A, Hirzinger G (2007) An analytical method for the planning of robust assembly tasks of complex shaped planar parts. In: IEEE International Conference on Robotics and Automation (ICRA), Rome, Italy, pp 317–323

Ude A (1999) Filtering in a unit quaternion space for model-based object tracking. Robotics and Autonomous Systems 28(2-3):163–172

Ude A, Gams A, Asfour T, Morimoto J (2010) Task-specific generalization of discrete and periodic dynamic movement primitives. IEEE Transactions on Robotics 26(5):800–815

Villani L, De Schutter J (2008) Force control. In: Siciliano B, Khatib O (eds) Springer Handbook of Robotics, Springer, Berlin, Heidelberg, pp 161–185

Whitney DE (1969) Resolved motion rate control of manipulators and human prostheses. IEEE Transactions on Systems, Man, and Cybernetics MMS-10(2):47–53

Whitney DE, Nevins JL (1979) What is the Remote Center Compliance (RCC) and what can it do? In: International Symposium on Industrial Robots (ISIR), Washington, DC

Xiao J (1997) Goal-contact relaxation graphs for contact-based fine motion planning. In: IEEE International Symposium on Assembly and Task Planning (ISATP), Marina del Rey, California, pp 25–30

Yamashita T, Godler I, Takahashi Y, Wada K, Katoh R (1991) Peg-and-hole task by robot with force sensor: Simulation and experiment. In: International Conference on Industrial Electronics, Control and Instrumentation (IECON), Kobe, Japan, pp 980–985

Yoshikawa T (2000) Force control of robot manipulators. In: IEEE International Conference on Robotics and Automation (ICRA), San Francisco, CA, pp 220–226

Yun SK (2008) Compliant manipulation for peg-in-hole: Is passive compliance a key to learn contact motion? In: IEEE International Conference on Robotics and Automation (ICRA), Pasadena, California, pp 1647–1652