2011 IEEE/RSJ International Conference on
Intelligent Robots and Systems
September 25-30, 2011. San Francisco, CA, USA

Online movement adaptation based on previous sensor experiences

Peter Pastor, Ludovic Righetti, Mrinal Kalakrishnan, and Stefan Schaal

Abstract— **Personal robots can only become widespread if they are capable of *safely* operating among humans. In uncertain and highly dynamic environments such as human households, robots need to be able to instantly adapt their behavior to unforseen events. In this paper, we propose a general framework to achieve very contact-reactive motions for robotic grasping and manipulation. Associating stereotypical movements to particular tasks enables our system to use previous sensor experiences as a predictive model for subsequent task executions. We use dynamical systems, named Dynamic Movement Primitives (DMPs), to learn goal-directed behaviors from demonstration. We exploit their dynamic properties by coupling them with the measured and predicted sensor traces. This feedback loop allows for online adaptation of the movement plan. Our system can create a rich set of possible motions that account for external perturbations and perception uncertainty to generate truly robust behaviors. As an example, we present an application to grasping with the WAM robot arm.**

I. INTRODUCTION

Autonomous robotic grasping is one of the prerequisites for personal robots to become useful when assisting humans in households. However, autonomous robotic grasping remains a very challenging problem, especially in such dynamic, cluttered, and uncertain environments [1]. In fact, a growing portion of robotics research directly or indirectly deals with all aspects related to complex human environments. A typical approach to robotic grasping [2],[3] involves 1) computer vision methods to estimate the object pose, 2) methods to compute a goal grasp configuration, 3) motion planning algorithms to generate a collision free trajectory from the current configuration to this goal configuration, and 4) controllers that track this trajectory. Usually, this processing pipeline is repeated for each new grasp attempt, i.e. each attempt is treated as a unique problem that is solved on its own. More importantly, information from previous grasp attempts is often discarded. Such approaches underestimate the importance of experienced sensor information from previous trials.

Associating experienced sensor information with corresponding robot action is very important as it allows to generate a predictive model of sensor traces that can be used to improve grasping over time (i.e. new grasp attempts can be evaluated using information from similar past grasp attempts). It also allows early detection of failure conditions during task execution [4] and to adapt movement plans online to unforeseen perturbations.

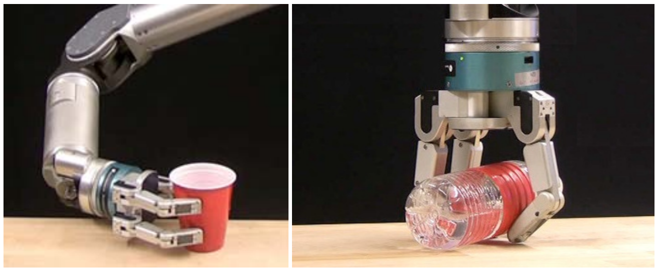

Fig. 1: The Barrett WAM arm grasping a cup and a bottle.

However, associating sensor information with a set of similar grasp attempts becomes very challenging when employing commonly used stochastic motion planning algorithms in step 3). For two very similar grasp scenarios these algorithms may return two very different approach trajectories resulting in two very different sensor experiences. Thus, transferring knowledge from previous trials may become impossible. Additionally, improving task performance over time and designing online adaptation algorithms become similarly challenging simply because the relation to previously successful trials is missing.

In contrast, learning a set of stereotypical movements that are applicable in certain grasping scenarios provides the basis to accumulate sensations of previous trials. Associating experienced sensor traces with stereotypical movements allows for building up a predictive model that can be used in future trials. Stereotypical movements facilitate the design of a system that can improve over time simply because new trials can be related to previous trials. Sensor traces from successful movements generate an implicit model of what the robot should *feel* for subsequent similar movements. Furthermore, such a predictive model can be systematically used to realize reactive behaviors that compensate for unexpected events. For example, during grasping, movement plans can be adapted online in case of early collisions with the object. The combination of reactive behaviors with stereotypical movements can create a rich set of possible motions that account for external perturbations and perception uncertainty to generate truly robust behaviors. In this work, we use the Dynamic Movement Primitive (DMP) framework to encode stereotypical movements. We extend this framework for online movement adaptation using sensory feedback. In particular, we present a control framework suitable for grasping, using DMPs with sensory feedback and nonlinear position and force control. First, we extend the DMP formulation

This research was supported in part by National Science Foundation grants ECS-0326095, IIS-0535282, IIS-1017134, CNS-0619937, IIS-0917318, CBET-0922784, EECS-0926052, CNS-0960061, the DARPA program on Advanced Robotic Manipulation, the Army Research Office, the Okawa Foundation, and the ATR Computational Neuroscience Laboratories.

The authors are with the CLMC Laboratory, University of Southern California, Los Angeles, USA. Stefan Schaal is also with the Max-Planck-Institute for Intelligent Systems, 72076 Tübingen, Germany. {pastorsa, kalakris, sschaal}@usc.edu, ludovic.righetti@a3.epfl.ch

978-1-61284-455-8/11/$26.00 ©2011 IEEE

for quaternion control, used to generate desired endeffector orientations. Then we propose a feedback law for DMPs that adapts the desired trajectories based on the difference between the forces experienced in a previous successful trial and the currently measured forces. We present a low-level position and force control system that seamlessly integrates with DMPs allowing for very reactive and compliant behaviors. Real robot experiments on a Barrett WAM show the advantages of this control framework for grasping under uncertainty. In our experiments we simulate errors in the vision system 1) and grasp pose estimation 2) by misplacing the object to be grasped while keeping the grasp goal configuration fixed. We show that open loop execution fails to grasp the object in most of the cases, while the proposed approach succeeds even for relatively large perception errors. The experiments successfully show that previous sensor information can be used to create robust grasping behaviors. Our control approach significantly improves the robustness of grasping while creating a very compliant control system.

II. RELATED WORK

In recent years, more and more robotic grippers and hands have become available with tactile and force sensing capabilities built-in. Since then, many approaches to grasping that incorporate sensor feedback have been proposed [5]. Hsiao [6] used tactile feedback to localize the position of a known object model for grasping. Platt [7] used force feedback to adjust an initial grasp until it is locally stable. Other approaches have been proposed that incorporate sensor feedback to account for positional uncertainty of the object [3], [8], [9]. These approaches use manually tuned thresholds to detect early collisions in order to trigger a replanning phase for subsequent grasp attempts. Dollar et al. [10] uses a similar pre-scripted reactive behavior in combination with a passively compliant hand. This work shows that not only the grasp range can be increased, but also the forces applied to the object can be reduced.

Sensor information has also been used for task outcome prediction. Peters et al. [11] proposed a method to discriminate successful from unsuccessful trials from sensor recordings of a teleoperated robot. Pastor et al. [4] proposed a method to use sensor recordings from previous trials to early detect failure conditions in future trials.

Dynamic Movement Primitives (DMPs) [12] are a set of differential equations that can compactly represent high-dimensional control policies. Their attractor landscapes can be adapted by only changing a few parameters, making the DMP representation particularly suitable for Reinforcement Learning [4], [13], [14]. DMPs can be used to encode any type of trajectory, e.g. joint trajectories [12], [15], endeffector trajectories [16], [17], and task space trajectories including desired nullspace postures [4]. These dynamical systems have been furthermore exploited to account for external perturbations [12] and for obstacle avoidance [16]. Gams et al. [17] have used DMPs to teach a robot a rhythmic movement necessary for wiping the surface of a table.

III. DYNAMIC SYSTEMS FOR MOVEMENT GENERATION

A. Learnable nonlinear point attractor systems

DMPs have the following favorable features:

- Any movement can be efficiently learned and generated.
- The representation is translation and time invariant.
- Convergence to the goal state g is guaranteed.
- Temporal/spatial coupling terms can be incorporated.

DMPs consist of three components: a transformation system, a nonlinear function approximator, and a canonical system. The transformation system is an analytically well understood dynamical system with convenient stability properties that, when integrated, generates the desired behavior. The nonlinear function approximator ensures that arbitrary movements can be encoded, and the canonical system avoids the explicit time dependency of the primitive and provides temporal coupling across multiple degrees of freedom (DOFs).

In this paper, the realization of the transformation system presented in [16] has been chosen – this is a special variant of the original DMP formulation that added robustness towards some numerical characteristics of the original DMPs, but has only reduced (localized) generalization as global translational invariance cannot be guaranteed. A discrete movement is generated by integrating the following transformation system[1]:

$$\begin{aligned} \tau\dot{v} &= K(g-x) - D\,v - K(g-x_0)s + Kf(s) \quad (1)\\ \tau\dot{x} &= v \ , \end{aligned}$$

where x and v are position and velocity of the system, x_0 and g are the start and goal position, τ is a temporal scaling factor, K is a spring constant, and D a damping term. The nonlinear function f is defined as

$$f(s) = \frac{\sum_i \psi_i(s)\,\theta_i\,s}{\sum_i \psi_i(s)} \ , \qquad (2)$$

where $\psi_i(s) = \exp(-h_i(s-c_i)^2)$ are Gaussian basis functions, with center c_i and width h_i, and θ_i are adjustable parameters. The function f does not directly depend on time; instead, it depends on a phase variable s, which monotonically changes from 1 towards 0 during a movement and is generated by the canonical system given by:

$$\tau\,\dot{s} = -\alpha\,s \ , \qquad (3)$$

where α is a pre-defined time constant. Multiple DOFs are realized by maintaining a separate set of transformation systems Eq. (1) as well as separate forcing terms Eq. (2) for each DOF and synchronize them via the canonical system Eq. (3) which becomes the central clock.

[1] A different notation is used as in [12] to highlight the spring-like character of these equations.

1) Learning from observed behavior: To encode a recorded trajectory $x(t)$ into a DMP, the parameters θ_i in Eq. (2) are adapted such that the nonlinear function forces the transformation system to follow this recorded trajectory. For this adaptation, the following four steps are carried out: first, using the recorded trajectory $x(t)$, the derivatives $v(t)$ and $\dot{v}(t)$ are computed for each time step $t = 0, ..., T$. Second, the canonical system Eq. (3) is integrated and $s(t)$ evaluated. Third, using these arrays, $f_{\text{target}}(s)$ is computed based on Eq. (1), where x_0 and g are set to $x(0)$ and $x(T)$, respectively. Finally, finding the parameters θ_i that minimize the error criterion $J = \sum_s (f_{\text{target}} - f(s))^2$ is a linear regression problem, which can be solved efficiently.

2) Movement generation using DMPs: The transformation system of Eq. (1) is at equilibrium when the current position x is equal the goal position g and $\dot{v} = s = 0$. To trigger a movement and obtain a desired plan $(x_{\text{desired}}, v_{\text{desired}}, \dot{v}_{\text{desired}})$ the system needs to be setup such that the condition for the unique equilibrium point $((x, v, s) = (0, g, 0))$ is no longer satisfied. Therefore, the start position is set to the current position $(x_0 \leftarrow x)$, the goal position is set to the desired goal $(g \leftarrow g_{desired} \neq x)$ and the canonical system is reset by setting $s \leftarrow 1$. The desired attractor landscape is obtained through plugging the learned set of parameters $\boldsymbol{\theta}$ into the nonlinear function f and the desired movement duration is adjusted using τ. The desired movement plan is obtained by integrating the canonical system, i.e. evaluating $s(t)$, computing the nonlinear forcing term $f(s)$ and integrating the transformation system.

B. DMPs for quaternion control

In control, the use of unit quaternions to represent rotations in $\mathbb{R}^3$ is very convenient [18]. Unit quaternions allow for non-singular representations of rotations and are easier to compose than for example Euler angles. Moreover, compared to rotation matrices, they are more compact and numerically stable. It is therefore desirable to have DMPs use a quaternion representation for orientations.

However, using 4 separate transformation systems Eq. (1) to represent an orientation trajectory generates quaternions that are not of unit length. Such a method is mathematically incorrect and requires an ad-hoc post-normalization step. Furthermore, it is difficult to interpret quaternion derivatives in terms of physical quantities. Indeed, the group of spatial rotation $SO(3)$ is a 3-dimensional manifold and therefore each of its element has a tangent space that is equivalent to $\mathbb{R}^3$. It is therefore unnecessarily complicated to use 4 derivatives to represent a 3-dimensional vector space.

In the following we propose a DMP formulation that guarantees the generation of unit quaternions by the dynamic system. Furthermore, the new integration rule offers a more intuitive representation of the derivatives allowing for tight integration in feedback control loops. The orientation error between two quaternions is defined as

$$\mathbf{e}_o(\{\eta_1, \mathbf{q}_1\}, \{\eta_2, \mathbf{q}_2\}) = \eta_1 \mathbf{q}_2 - \eta_2 \mathbf{q}_1 - \mathbf{q}_1^{\times} \mathbf{q}_2 \quad , \qquad (4)$$

where $\{\eta, \mathbf{q}\}$ is a unit quaternion with $\eta \in \mathbb{R}$ and $\mathbf{q} \in \mathbb{R}^3$ and $\mathbf{q}^{\times}$ is the skew-symmetric matrix given by

$$\mathbf{q}^{\times} = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{bmatrix} . \qquad (5)$$

Note that the orientation error $\mathbf{e}_o(\{\eta_1, \mathbf{q}_1\}, \{\eta_2, \mathbf{q}_2\}) \in \mathbb{R}^3$ can be interpreted as an error in angular velocities [18].

Here, we use this error computation to compose a single transformation system which generates unit quaternions $\mathbf{q}$, as well as angular velocities $\boldsymbol{\omega}$ and angular accelerations $\dot{\boldsymbol{\omega}}$. Replacing the error computation in Eq. (1) leads to

$$\begin{aligned} \tau \, \dot{\boldsymbol{\omega}} \;\; &= \;\; -\mathbf{K} \cdot \mathbf{e}_o(\mathbf{g}, \{\eta, \mathbf{q}\}) - \mathbf{D} \cdot \boldsymbol{\omega} \\ &\quad + \mathbf{K} \cdot \mathbf{e}_o(\mathbf{g}, \{\eta_0, \mathbf{q}_0\}) \; s + \mathbf{K} \cdot \mathbf{f}(s) \qquad (6) \end{aligned}$$

$$\tau \begin{bmatrix} \dot{\eta} \\ \dot{\mathbf{q}} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & -\boldsymbol{\omega}^{\times} \end{bmatrix} \begin{bmatrix} \eta \\ \mathbf{q} \end{bmatrix} . \qquad (7)$$

Note, the scalars in Eq. (1) have been replaced by vectors. See [18] for more details on quaternion propagation. The proposed quaternion integration rule guarantees the generation of normalized unit quaternions. Furthermore, it ensures that the generated quaternion orientations and corresponding angular velocities and accelerations are differential compatible. Moreover, this new integration rule facilitates a tight integration of the dynamical system into control loops, as described in the following subsection.

C. Online trajectory generation using sensory feedback

The main idea of our approach is to online modify the desired trajectory generated by the DMP using sensor information from previous task executions. Here, the concept of stereo typical movements plays a key role: Similar tasks afford similar movement plans. This simple yet profound insight enables our system to accumulate past sensor experiences in form of sensor traces and use them in future similar scenarios, for example for failure detection [4]. These sensor traces compose a growing associative memory relating task execution with sensor experiences. Predictive models of how things should *feel* during task execution are particularly relevant during contact interactions of the robot with its environment. Our approach uses this information to online modify the desired trajectory, i.e. the movement plan, such that the measured sensory experience remains close to the expected one. Given information of force, torque, or pressure, the corresponding desired generalized forces acting on the task variables can be computed. For example, a force/torque sensor at the wrist, provides a direct relation between sensing and endeffector accelerations. The desired acceleration can be modified to compensate for sensing errors.

The general feedback function is given by

$$\boldsymbol{\zeta} = \mathbf{K}_1 \mathbf{J}_{sensor}^T \mathbf{K}_2 (\mathbf{F} - \mathbf{F}_{des}) \ , \qquad (8)$$

where $\mathbf{J}_{sensor}$ is the Jacobian of the task controlled by the movement primitives with respect to the sensors, $\mathbf{F}$ are the generalized forces read from the sensors and $\mathbf{F}_{des}$ are the corresponding desired forces, that have been acquired in previous task executions. $\mathbf{K}_1$ and $\mathbf{K}_2$ are, potentially time-varying, positive definite gain matrices. The feedback is added to the transformation system in Eq. (1) as follows

$$\begin{aligned} \tau\dot{v} &= K(g-x) - D\,v - K(g-x_0)s + Kf(s) + \zeta \\ \tau\dot{x} &= v \ . \end{aligned} \tag{9}$$

Similarly feedback to the quaternion DMPs is added directly in Eq. (6). For example, in the case of using DMPs to control the hand position and orientation and considering a force/torque sensor located at the wrist of a manipulator, the Jacobian of the task becomes $\mathbf{J}_{sensor} = \mathbf{I}_{6\times 6}$ and $\mathbf{F}$ is the 6-dimensional reading from the sensor.

Remark The gain matrix $\mathbf{K}_1$ defines the sensitivity of the task variables with respect to predicted forces. Each dimension (position and orientation) can be weighted differently. For example, we can imagine a task where one direction should adapt very quickly to sensing while the other directions should be more stiff. The gain matrix $\mathbf{K}_2$ defines the relative sensitivity to different sensors, depending for example on their resolution or adequacy to the task. We would imagine that such a gain matrix will be related to the known variance of the desired forces acquired during learning (see [4], [19]).

IV. POSITION AND FORCE CONTROL

We separate the low-level controller into two distinct parts: a controller for the endeffector, i.e. the hand, and a controller for the fingers.

A. Hand position and force controller

1) Endeffector position controller: In order to track the desired positions and orientation generated by the DMP we use a velocity-based operational space controller together with an inverse dynamics law and feedback error compensation in joint space (see [20] for more details). The control law is written as

$$\boldsymbol{\tau}_{arm,p} = \mathbf{M}\ddot{\mathbf{q}}_d + \mathbf{h} + \mathbf{K}_p(\mathbf{q}_d - \mathbf{q}) + \mathbf{K}_d(\dot{\mathbf{q}}_d - \dot{\mathbf{q}}) \tag{10}$$

with

$$\dot{\mathbf{q}}_d = \mathbf{J}^{+}\Big(\dot{\mathbf{x}}_d + \mathbf{K}_x(\mathbf{x}_d - \mathbf{x})\Big) + \mathbf{K}_{post.}\big(\mathbf{I} - \mathbf{J}^{+}\mathbf{J}\big)(\mathbf{q}_{post.} - \mathbf{q}), \tag{11}$$

where $\mathbf{M}$ is the rigid-body inertia matrix of the arm, $\mathbf{q}$ and $\mathbf{q}_d$ are the vectors of measured and desired joint angles, $\mathbf{x}$ and $\mathbf{x}_d$ are the measured and desired endeffector position and orientation, $\mathbf{h}$ is the vector of Coriolis, centrifugal, and gravitational forces, $\mathbf{K}_p, \mathbf{K}_d, \mathbf{K}_x$, and $\mathbf{K}_{post.}$ are (diagonal) gain matrices and $\boldsymbol{\tau}_{arm,p}$ is the vector of torques. $\mathbf{J}$ is the endeffector Jacobian, $\mathbf{J}^{+}$ denotes the Moore-Penrose generalized inverse and $\mathbf{q}_{post.}$ is the vector of default posture optimized in the nullspace of the endeffector motion. The desired acceleration $\ddot{\mathbf{q}}_d$ and desired position $\mathbf{q}_d$ are obtained by numerical differentiation and integration of the desired velocity $\dot{\mathbf{q}}_d$.

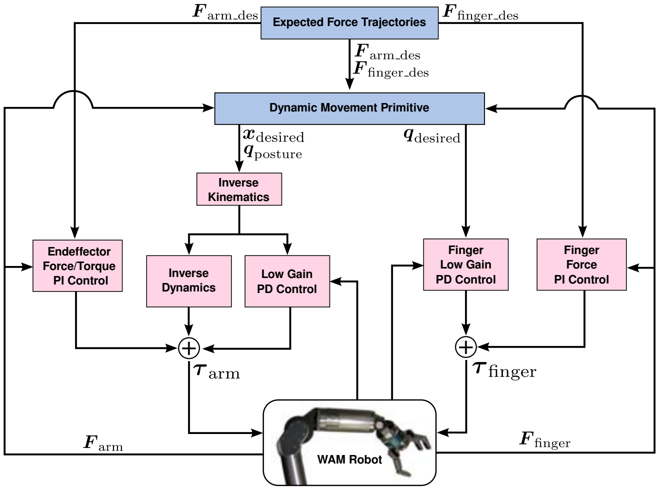
$\boldsymbol{F}_{\mathrm{arm\_des}}$
Expected Force Trajectories
$\boldsymbol{F}_{\mathrm{finger\_des}}$
$\boldsymbol{F}_{\mathrm{arm\_des}}$
$\boldsymbol{F}_{\mathrm{finger\_des}}$
Dynamic Movement Primitive
$\boldsymbol{x}_{\mathrm{desired}}$
$\boldsymbol{q}_{\mathrm{posture}}$
$\boldsymbol{q}_{\mathrm{desired}}$
Inverse Kinematics
Endeffector Force/Torque PI Control
Inverse Dynamics
Low Gain PD Control
Finger Low Gain PD Control
Finger Force PI Control
$\boldsymbol{\tau}_{\mathrm{arm}}$
$\boldsymbol{\tau}_{\mathrm{finger}}$
$\boldsymbol{F}_{\mathrm{arm}}$
WAM Robot
$\boldsymbol{F}_{\mathrm{finger}}$

Fig. 2: Diagram of the proposed control architecture.

Remark This controller uses differential inverse kinematics to generate desired joint position, velocities and accelerations (Eq. (11)) and a standard inverse dynamics controller with PD error feedback to achieve the desired trajectories (Eq. (10)). Inverse dynamics control allows to reduce significantly the error feedback gains for compliant control while ensuring high tracking performances.

For the position control we assume that there are no contacts with the environment, therefore our inverse dynamics control law does not take into account possible external forces. The contacts with the environment are dealt with using the following force controller.

2) Endeffector force controller: The tracking of desired contact forces is achieved with a PI controller

$$\begin{aligned} \boldsymbol{\tau}_{arm,f} \quad &= \quad -\mathbf{J}^{T}\Big(\mathbf{F}_{arm\_des} - \mathbf{F}_{arm} \\ &\qquad + \mathbf{K}_i \int_{t-\Delta t}^{t} \left(\mathbf{F}_{arm\_des} - \mathbf{F}_{arm}\right)\, dt\Big), \end{aligned} \tag{12}$$

where $\mathbf{F}_{arm}$ and $\mathbf{F}_{arm\_des}$ are the measured and desired forces at the endeffector, $\mathbf{K}_i$ is a (diagonal) positive definite gain matrix and Δt represents the time-window during which the force error is integrated. Note that we use unitary gains for the proportional case since it exactly eliminates the sensed force and replace it by the desired one assuming a perfect rigid body system. The integral controller will compensate for steady-state errors during contact.

3) Combined position and force controller: The final command sent to the arm joints is computed according to

$$\boldsymbol{\tau}_{arm} = \boldsymbol{\tau}_{arm,p} + \boldsymbol{\tau}_{arm,f} \ . \tag{13}$$

By choosing an appropriate desired position and force these controllers act together to achieve the desired task. Our design of the feedback at the DMP level create force-consistent desired trajectories. The online adaptation of the DMP, thus, allow us to simply use the addition of both position and force controller.

Remark The resulting controller is simple but is not as general as an impedance controller would be. However it has the advantages that we do not need to explicitly design a desired impedance behavior in our control. Indirectly, the choice of (potentially time-varying) gains related to the DMP adaptation and to the force controller will define a desired impedance for the controller. This controller turns out to give us sufficiently good performance in practice. Note that we would not be able to use a hybrid force/position controller since we don't want to decouple the directions of control for the position and the force.

B. Finger position and force controller

The fingers are controlled with a position PD controller and a force PI controller. The force readings are obtained from the strain gages sensors located in the second joint of each finger. The spread between the left and right fingers is only controlled in position. The total torque command for the fingers is computed according to

$$\boldsymbol{\tau}_{finger} = \mathbf{PD}_{position} + \mathbf{PI}_{force} \quad . \qquad (14)$$

Remark This controller renders the fingers compliant even if they are non-backdrivable (which is the case in our experiments). It complements the finger DMP adaptation to ensure a fast response to perturbations.

An overview of the presented control architecture is shown in Fig. 2.

V. LEARNING FROM PREVIOUS EXPERIENCES

Our approach aims to generate a rich set of movement primitives that allow online adaptation of the desired behavior based on *external* forces and torques. To acquire new motor skills we propose a two staged approach: First, use imitation learning to bootstrap the learning process and encode all kinematic information. Second, execute the learned behavior in open loop mode, i.e. setting the feedback terms $\zeta = 0$ in Eq. (9), and record the experienced *internal* forces and torques. These recorded force and torque trajectories resemble a predictive model of expected sensor readings *without* external perturbations. Thus, the desired kinematic trajectory will only adapt if there is a deviation between the measured and the predicted forces and torques. Furthermore, accurate modeling of the inertia parameters of the endeffector, for example hand and fingers, can be omitted since they are taken into consideration during the open loop execution. Our approach therefore works seamlessly when additional weight needs to be compensated, for example when lifting a cup.

VI. EXPERIMENTAL RESULTS

A. Experimental setup

The experiments were conducted using the 7 DOF WAM robot arm and the 4 DOF Barrett Hand BH-280, both built by Barrett Technology, Inc. Additionally, we used the force/torque sensor mounted on the wrist of the arm and the joint-torque sensors embedded in the knuckles of the 3 fingers to close the force control loops (Fig. 2). The overall position and force control loop including reading all sensor values and sending desired torques runs at 300 Hz.

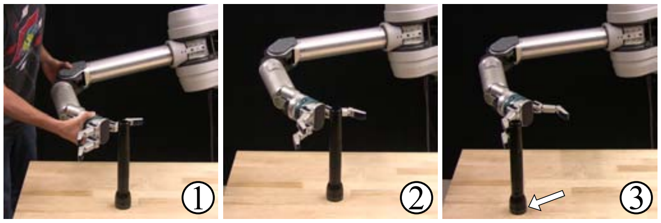

Fig. 3: Proposed method: (1) Use imitation learning to learn movement plan, (2) execute learned movement and record sensor information, (3) use experienced sensor information to react to unforeseen perturbations.

In our grasping experiments, we followed the procedure sketched in Fig. 3. The grasping DMP was learned through kinesthetic teaching, i.e. guiding the gravity compensated WAM robot from an initial position to an appropriate grasp location. From this demonstration, we extracted the endeffector trajectory and learned a DMP in Cartesian space using the new quaternion formulation described in III-B. Additionally, the desired finger joint positions were learned from a minimum jerk trajectory that was generated to ensure synchronous finger closing. Thus, we used kinesthetic teaching to learn a single DMP encoding all kinematics information, i.e. the desired endeffector position and orientation, the desired default posture[2] and the desired finger trajectories. To obtain a predictive model of the forces, as described in V, we executed the learned DMP and recorded the forces and torques sensed at the wrist of the WAM robot arm. These force profiles resemble the *default* sensor experience for this particular grasp (see Fig. 2).

We tested the proposed method in three tabletop grasping experiments: Grasping a cup (Fig. 1), an upright standing maglite flashlight (Fig. 3) and a lying bottle of water (Fig. 1). All objects have been misplaced in order to simulate uncertainty.

B. Results

A typical behavior of the proposed approach while grasping a misplaced red cup from the side as depicted in Fig. 1 is shown in Fig. 4. The results show that the robot significantly adapted its motion (up to 12 cm in position and 20 degrees in orientation) resulting in a successful grasp[3].

In order to quantify the improvements in grasping under uncertainty more rigorously, we performed a systematic test for grasping a maglite flashlight. To simulate uncertainty, we placed the maglite on a grid composed of 4x4 cm squares while keeping the goal grasp configuration fixed. We considered three different control modes: (a) open loop, (b) force control without DMP adaptation, and (c) force control with DMP adaptation of the desired movement plan. A grasp attempt was marked successful if the robot achieved a power grasp. If the flashlight was knocked over, for example due to early contact, the grasp attempted was marked unsuccessful. The result of the systematic test is shown in Fig. 5.

[2] The default posture is used in Eq. (11) to resolve the redundancies of the 7 DOF WAM robot.

[3] The highlights of each grasping experiment are shown in the video supplement and http://www.youtube.com/watch?v=ZWkF2peI1qw

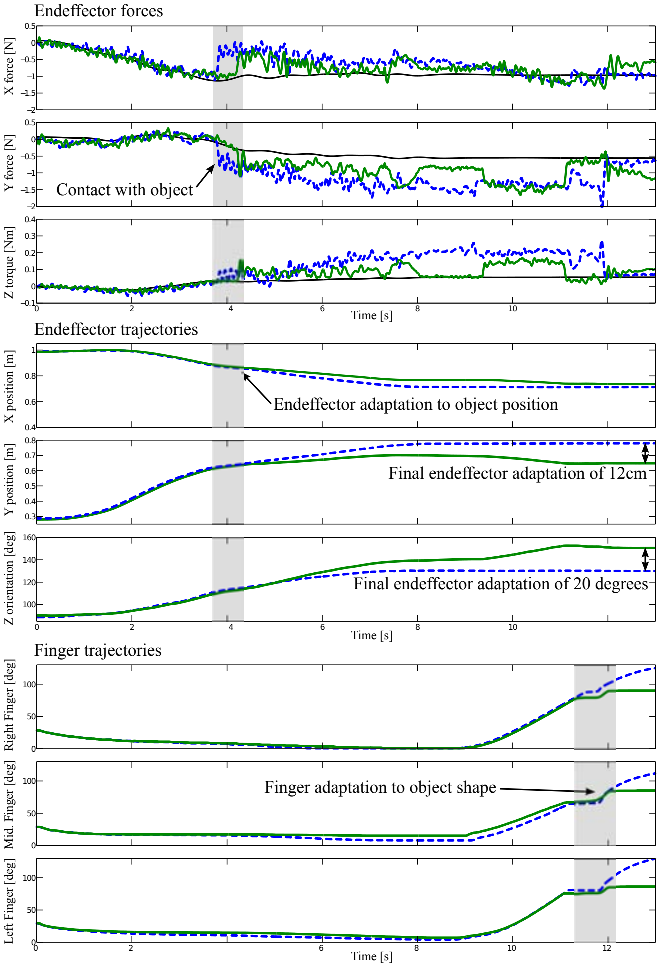

Fig. 4: Typical behavior obtained from grasping a misplaced cup. The top three plots show the endeffector forces and torque in the horizontal plane, the middle three plots show the endeffector positions and orientation in the same plane, and the bottom three plots show the finger joint trajectories. The green lines were obtained from experiments with DMP adaptation, the blue dashed lines were obtained from open loop execution. The black lines in the top three plots correspond to the expected desired forces recorded from a previous successful grasp. In the beginning of the movement the closed loop endeffector trajectories remain close to the open loop endeffector trajectories. After unexpectedly touching the misplaced cup (left gray region around 4 seconds) the closed loop controller adapts the endeffector positions and orientation leading to a successful grasp. The finger joint trajectories are adapted to the shape of the cup (right gray region around 12 seconds).

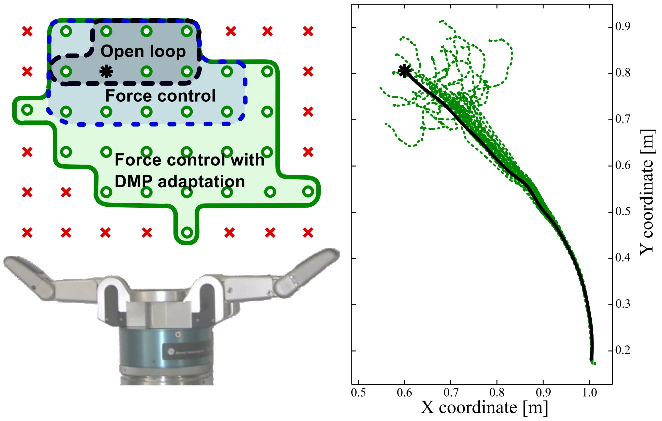

Fig. 5: Result of each grasping attempt after misplacing the flashlight (left). The grid is 28 cm large and 20 cm wide. There is 4 cm between each adjacent crosses. The region in which open loop (black dashed) execution succeeds is rather small. Adding force control (blue dashed) increases the region. However, only when the presented DMP adaptation is used, the region of successful grasps increases significantly. The thick black line shows the open loop trajectory and green dashed lines show how those trajectories were adapted after touching the object (right).

As shown in Fig. 5, the region of successful grasps is increased if the force control loop is closed. However, only if both, the force control loop as well as the DMP feedback loop are closed the region of successful grasps is increased significantly. Even if the object was misplaced by up to 23 cm, the robot was able to gently give in (without knocking over the flashlight) and successfully grasp the flashlight. Snapshots from this experiment are shown in Fig. 6.

The same approach was successful in grasping a water bottle lying on the table as shown in Fig. 1. This experiment suggest that our approach generalizes to a variety of grasp scenarios requiring important contact interaction and compliant behavior. In this particular experiment, the execution of the learned movement in open loop mode to record the endeffector forces (step 2 in Fig. 3) has been done without the table. Indeed, we decided to not execute the movement without the force controllers because of the danger of breaking the fingers of the robot due to missing compensation when touching the table. In contrast, our results with the force controllers and the DMP adaptation show that our approach is safe (see also video-supplement). Unexpected collisions with the table are compensated in a controlled fashion. This experiment emphasizes the compliant properties of the controller and its ability to generate feasible motion when in contact with the environment. We expect to learn such desired forces in the future, as was recently shown in [19].

VII. Conclusion and Future Work

In this paper we proposed a sensory feedback law for DMPs to realize online movement adaptation. Alongside, we proposed a mathematically correct DMP formulation for orientation trajectories represented as unit quaternions. We highlighted the importance of stereotypical movements and proposed a general framework that uses sensor information from previous successful trials to systematically compensate for unforeseen perturbations in future trials. The presented approach has been integrated with a low-level position and force control system. The experiments show that our control system is compliant and reactive enough to sense a misplaced upright standing flashlight. Furthermore, our system was able to online adapt its movement plan without knocking over the flashlight. Our approach significantly increased the region of successful grasps. It also allowed to create very compliant behaviors for contact interaction with the environment as we demonstrated in the water bottle grasping that required contact with the table.

Future work will focus on generating libraries of motor skills. Using DMPs as representation for stereotypical

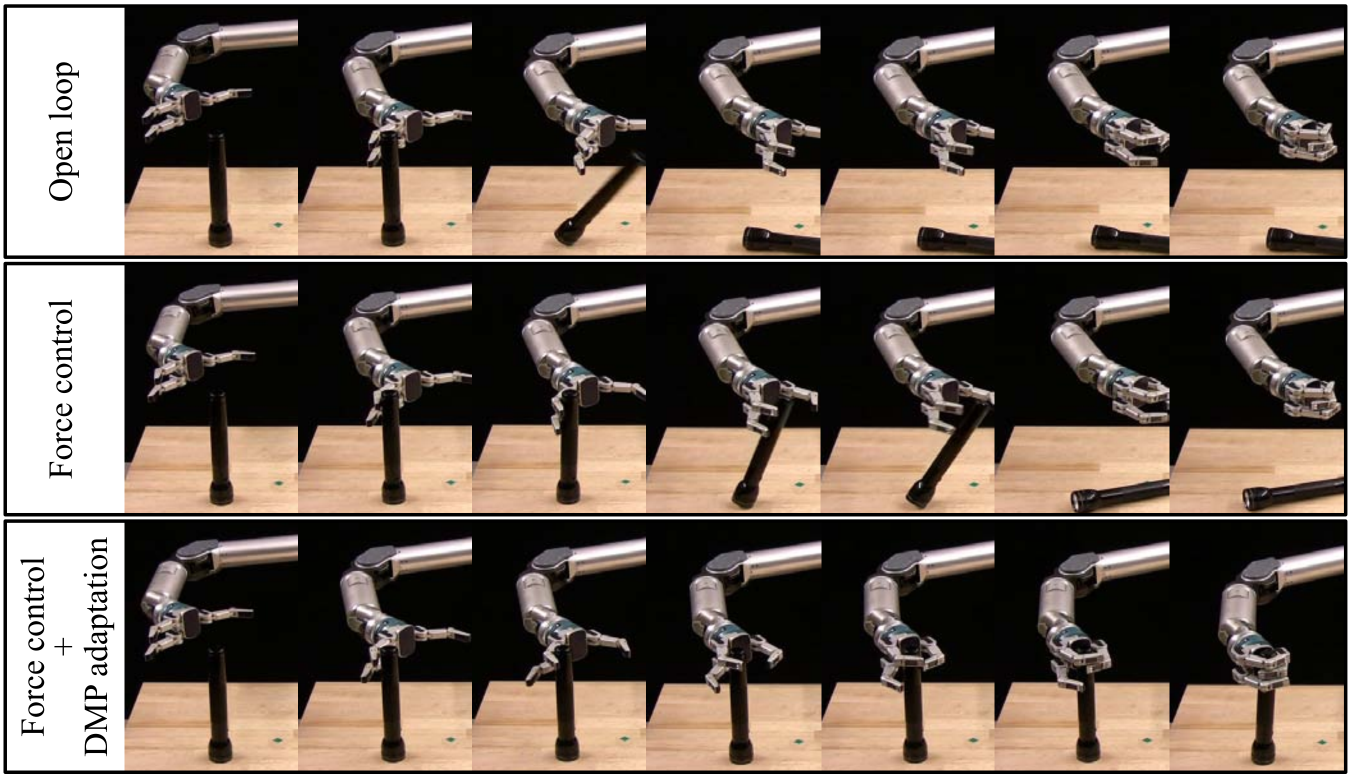

Fig. 6: Snapshots of the flashlight grasping experiment. Uncertainty is simulated by placing the flashlight off the target (green marker on table) while keeping movement goal fixed. Open loop execution (top row) fails immediately. Including force control to servo the desired force allows for slight adaptation, however, the tracking error overrules the force controller and the grasp fails as well (middle row). Only if the DMP adapts the desired trajectory, the grasp succeeds (bottom row).

movements, we will work towards integrating the presented approach with Reinforcement Learning to optimize position as well as force policies [19] and task outcome prediction methods [4]. We will also work towards developing methods that reason about which sensor traces are to be expected for a given task. Our hope is that such methods can even infer new sensor traces for new tasks from possibly multiple previously experienced sensor traces.

REFERENCES

[1] C. Kemp, A. Edsinger, and E. Torres-Jara, "Challenges for robot manipulation in human environments," *Robotics Automation Magazine, IEEE*, vol. 14, no. 1, pp. 20–29, 2007.

[2] S. Srinivasa, D. Ferguson, C. Helfrich, D. Berenson, A. Collet, R. Diankov, G. Gallagher, G. Hollinger, J. Kuffner, and M. VandeWeghe, "Herb: A home exploring robotic butler," *Autonomous Robots*, 2009.

[3] M. Ciocarlie, K. Hsiao, E. G. Jones, S. Chitta, R. B. Rusu, and I. A. Sucan, "Towards reliable grasping and manipulation in household environments," in *Intl. Symposium on Experimental Robotics*, 2010.

[4] P. Pastor, M. Kalakrishnan, S. Chitta, E. Theodorou, and S. Schaal, "Skill Learning and Task Outcome Prediction for Manipulation," in *Proc. of the IEEE Intl Conf. on Robotics and Automation*, 2011.

[5] L. Natale and E. Torres-Jara, "A sensitive approach to grasping," in *Sixth Intl Conf. on Epigenetic Robotics*, 2006.

[6] K. Hsiao, "Relatively robust grasping," Ph.D. dissertation, Massachusetts Institute of Technology, 2009.

[7] R. Platt, "Learning Grasp Strategies Composed of Contact Relative Motions," in *7th IEEE-RAS International Conference on Humanoid Robots*, 2007, pp. 49–56.

[8] J. Felip and A. Morales, "Robust sensor-based grasp primitive for a three-finger robot hand," in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems*, 2009, pp. 1811–1816.

[9] K. Hsiao, S. Chitta, M. Ciocarlie, and E. G. Jones, "Contact-reactive grasping of objects with partial shape information," in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems*, 2010.

[10] A. M. Dollar, L. P. Jentoft, J. H. Gao, and R. D. Howe, "Contact sensing and grasping performance of compliant hands," *Autonomous Robots*, vol. 28, pp. 65–75, 2010.

[11] R. Peters, R. Bodenheimer, and O. Jenkins, "Sensory-Motor Manifold Structure Induced by Task Outcome: Experiments with Robonaut," in *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, 2006, pp. 484–489.

[12] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Learning Attractor Landscapes for Learning Motor Primitives," in *Advances in Neural Information Processing Systems*, 2003, pp. 1547–1554.

[13] J. Kober and J. Peters, "Learning Motor Primitives for Robotics," in *Proc. of the IEEE Intl Conf. on Robotics and Automation*, 2009.

[14] F. Stulp, E. Theodorou, J. Buchli, and S. Schaal, "Learning to grasp under uncertainty," in *Proc. of the IEEE Intl Conf. on Robotics and Automation*, 2011.

[15] J. Kober and J. Peters, "Policy search for motor primitives in robotics," in *Advances in Neural Information Processing Systems*, 2008.

[16] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and Generalization of Motor Skills by Learning from Demonstration," in *Proc. of the IEEE Intl Conf. on Robotics and Automation*, 2009.

[17] A. Gams, M. Do, A. Ude, T. Asfour, and R. Dillmann, "On-line periodic movement and force-profile learning for adaptation to new surface," in *10th IEEE-RAS International Conference on Humanoid Robots*, 2010.

[18] J. Yuan, "Closed-loop manipulator control using quaternion feedback," *IEEE Journal of Robotics and Automation*, vol. 4, no. 4, pp. 434 –440, 1988.

[19] M. Kalakrishnan, L. Righetti, P. Pastor, E. Theodorou, and S. Schaal, "Learning Force Control Policies for Compliant Manipulation," in *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems*, 2011.

[20] J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal, "Operational space control: a theoretical and emprical comparison," *International Journal of Robotics Research*, vol. 27, no. 6, pp. 737–757, 2008.