ICT Call 7
ROBOHOW.COG
FP7-ICT-288533

Deliverable D5.3:

Publication on learned haptics interaction

February 1, 2015

Project acronym:	ROBOHOW.COG
Project full title:	Web-enabled and Experience-based Cognitive Robots that Learn Complex Everyday Manipulation Tasks
Work Package:	WP 5
Document number:	D5.3
Document title:	Publication on learned haptics interaction
Version:	1.0
Delivery date:	January 31st, 2015
Nature:	Report
Dissemination level:	Public
Authors:	Brahayam Ponton, Ana-Lucia Pais Ureche, Nadia Figueroa, Miao Li and Aude Billard (EPFL); Francois Keith and Abderrahmane Kheddar (CNRS); Kaiyu Hang and Danica Kragic (KTH)

The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement n°288533 ROBOHOW.COG.

Contents

Summary

This deliverable reports on work performed as part of WP5 during the third year of the project. The work addressed Task 5.3. on *Learning adaptive stiffness control that has the desired effects* and Task 5.4 on *Learning to assess grasp stability*. It stems from collaborations between EPFL and CNRS and between EPFL and KTH. The report is divided into 3 chapters, whose contributions we summarize briefly next.

In Chapter 1 and 2, we report on on two strategies to learn adaptive stiffness to achieve the desired effect on the world.

First, in Chapter 1, we present a method for performing adaptive force control at the arm level, that is able to estimate on-line friction, and generate a control law to overcome static friction and properly transition to dynamic friction, for safely achieving a task goal. We test this approach on a drawer opening task, and show generalization with respect to the weight and type of the drawer. This work is the result of a collaboration between EPFL and CNRS. It was validated on two different robotic platforms: the Kuka LWR at EPFL and the HRP4 at CNRS.

Second, in Chapter 2 we present an extension of the work reported during the second year of the project to automatically segment the task and extract constraints for the task. This year, we emphasized the modeling of the change in impedance parameters and their effect on inducing deformation of objects. We illustrate this in a cooking task, namely rolling a dough. This work is currently being implemented onto the Boxy platform at UNIB and will be showcased during the review meeting.

Finally, in Chapter 3 we report on recent progresses done for learning impedance control at the level of the hand, in order to achieve stable grasps. The method extracts haptic patterns recorded through touch-sensitive sensors located at the fingertips during human demonstration and learns a mapping between these tactile responses and the positioning of the fingers on the object to ensure stability. This approach is designed to offer online adaptation to perturbations by adjusting the position of the fingers or the strength of the force applied by the fingers to re-stabilize the object. This work contributes to *Task 5.3 Learning adaptive stiffness control that has the desired effects* and *Task.5.4 Learning to assess grasp stability*. The work builds upon the two last years of work by EPFL on adaptive stiffness control and KTH on assessing grasp stability. Part of this work had already been reported in year 2, deliverable D5.2. Here, we include in annexes this year's publications at ICRA(Appendix C) and IROS(Appendix D). Additionally this report also includes the joint work with KTH (WP4) on grasp adaptation, we integrate the work of adaptive stiffness control with the grasp planning to maintain the grasp stability. This work aims to address particular the challenge of grasp adaptation in Task 5.4, i.e.,"Sensory knowledge affects

the success of grasping process both in the planning stage (before a grasp is executed) and during the execution of the grasp (closed-loop on-line control) and we will study both of these aspects". A detailed report for this work is included in Appendix E.

Chapter 1

Adaptive force control for opening/closing a drawer

In this work, we focus on designing a controller capable of dealing with varying task conditions (e.g. friction, mass, etc.) that can be learned/updated during execution, resulting in a compliant behavior coveted in robotic applications. In particular, we are interested in extracting a control principle that allows to successfully perform a task that transitions between static and dynamic friction, such as opening a bottle, adjusting a light bulb, opening or closing a drawer/tray, etc. As a first application, we focused on opening/closing of both a drawer and a printer's tray (as shown in Figure 1.1).

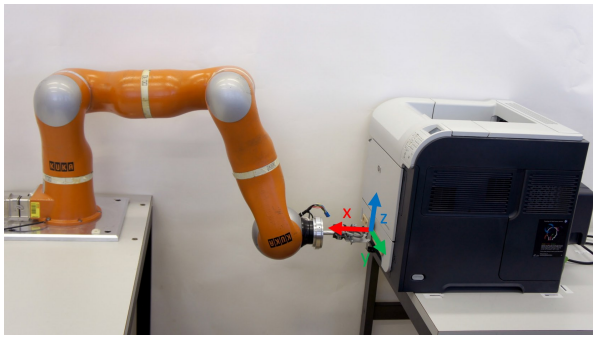


Figure 1.1: The Robotic Setup for a tray opening/closing task for the proposed control system.

The proposed control system is illustrated in Figure 1.2. The robot interacts with the printer's tray, represented by the *Disturbance Load & Friction* block. The *Delay* block represents delays present in real world applications. The LQR controller is used as a *Policy* that assumes no friction. It receives as input the current state (current position x and current velocity $\dot{x}$ of the end-effector) and outputs the desired state (desired position x_d, desired velocity $\dot{x}_d$ and desired acceleration $\ddot{x}_d$). Our proposed *Adaptive Controller* block, adaptively builds a model of the mass and friction to be used for feed-forward compensation, controls the robot's compliance to compensate for unknown/un-modeled dynamics, all the while ensuring exponential stability with non-linear damping.

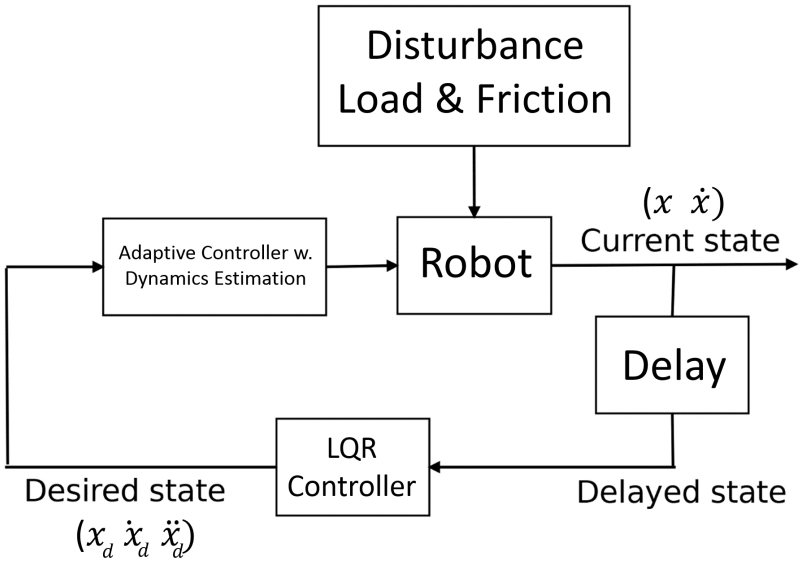


Figure 1.2: Diagram of the control system described in detail in the paragraph above.

Controller Derivation: We simplify the task of opening/closing a printer's tray to a 1D problem, namely controlling solely for the push/pull direction of the tray in task space (i.e. end-effector). This reduces our system dynamics from the classical equation of motion for robots in task space to the following equation:

$$m_x\ddot{x} + F_{fr}(\dot{x}) = u - F_{load} \tag{1.1}$$

where $F_{fr}(\dot{x})$ is the friction in the push/pull direction (x direction as shown in Figure 1.1). $F_{load} = m_{obj}\ddot{x}$ represents the load force from the printer tray's mass m_{obj}. $\ddot{x}$ is the Cartesian acceleration in the controlled direction. m_x is an estimate of the end-effector mass and u is the command that the robot should execute to compensate friction and open the printer's tray. We can simplify this equation by considering $m = m_x + m_{obj}$, which accounts for unknown object mass and uncertain manipulator mass. Hence, our system dynamics can now be expressed as:

$$m\ddot{x} = u - F_{fr}(\dot{x}) \tag{1.2}$$

We then formulate a control law that will compensate for inertial and friction effects based on building a model of the estimated mass $\hat{m}$ and estimated friction $\hat{F}_{fr}$ and for unknown disturbances with a spring (K) - damper (D) system, as follows:

$$u = \hat{m}\ddot{x}_d + \hat{F}_{fr}(\dot{x}) - Ke - D\dot{e} \tag{1.3}$$

where $e = x - x_d$ and $\dot{e} = \dot{x} - \dot{x}_d$ are position and velocity errors, respectively.

Friction Compensation: With the goal of maximizing robustness in task execution and re-usability for different tasks and conditions, we chose to use a static friction compensation model (Ge et al, 2001), where friction $F_{fr}(\dot{x})$ can be approximated by a weighted sum of basis functions as $F_{fr}(\dot{x}) = \Psi_s(\dot{x})^T\theta$. Each basis function $\Psi_s(\dot{x})$ represents friction at different velocity conditions with true friction parameters θ. The estimated friction estimation is then:

$$\hat{F}_{fr}(\dot{x}) = \Psi_s(\dot{x})^T\hat{\theta} \tag{1.4}$$

Figure 1.3: Human demonstration of opening/closing a tray.

where $\hat{\theta}$ is the friction parameter estimate and $\Psi(\dot{x})$ is chosen as:

$$\Psi_s(\dot{x}) = \left[\text{sign}(\dot{x}), \dot{x}, \dot{x}|\dot{x}|, \sqrt{|\dot{x}|}\text{sign}(\dot{x})\right]^T \tag{1.5}$$

The different terms in the set of basis functions compensate for (i) Coulomb friction, (ii) viscous friction, (iii) drag friction, and (iv) a square root friction term for high contact pressure.

Nonlinear Damping: In order to improve the controller's convergence, we introduce a non-linear damping term $K_2\tilde{E}$ in our control law Eq.1.3. Considering the chosen friction compensation model, our proposed control law becomes:

$$u = \hat{m}\ddot{x}_d + \Psi_s(\dot{x})^T\hat{\theta} - Ke - K_2\tilde{E}\dot{e} \tag{1.6}$$

where K_2 is a gain parameter and $\tilde{E} = E - E_d$ is the energy difference between the current and desired state, derived from a Lyapunov function E which defines the energy level of the system at the current state:

$$E = \frac{1}{2}m\dot{e}^2 + \frac{1}{2}Ke^2 + \frac{1}{2}\frac{\tilde{m}^2}{\omega_m} + \frac{1}{2}\tilde{\theta}^T\Gamma^{-1}\tilde{\theta} \tag{1.7}$$

$\tilde{m}$ and $\tilde{\theta}$ are the estimation errors for the mass and friction parameters, respectively; ω_m and Γ are gain parameters. By setting the desired energy level to $E_d = 0$ and introducing the error dynamics equation (formulated by combining Eq. 1.2 and Eq.1.6), to the derivative of the energy difference $\dot{\tilde{E}}$, we obtain the following update rules for the mass $\dot{\hat{m}} = -\omega_m\ddot{x}_d\dot{e}$ and friction parameters $\dot{\hat{\theta}} = -\Gamma\Psi\dot{e}$, which simplify the energy difference derivative to $\dot{\tilde{E}} = -K_2\dot{e}^2\tilde{E}$ and consequently:

$$\tilde{E}(t) = \tilde{E}(0)\text{exp}(-K_2\dot{e}^2t), \tag{1.8}$$

which shows that the energy level decays exponentially. For a full derivation of the control law, update rules for the mass and friction estimates and stability proof refer to Appendix A.

1.1 Implementation on the Kuka LWR Robot

We used the KUKA LWR robot arm to demonstrate opening/closing of a printer's tray. A metallic piece was built to rigidly attach the robot's end-effector to the printer's tray (as shown in Figure

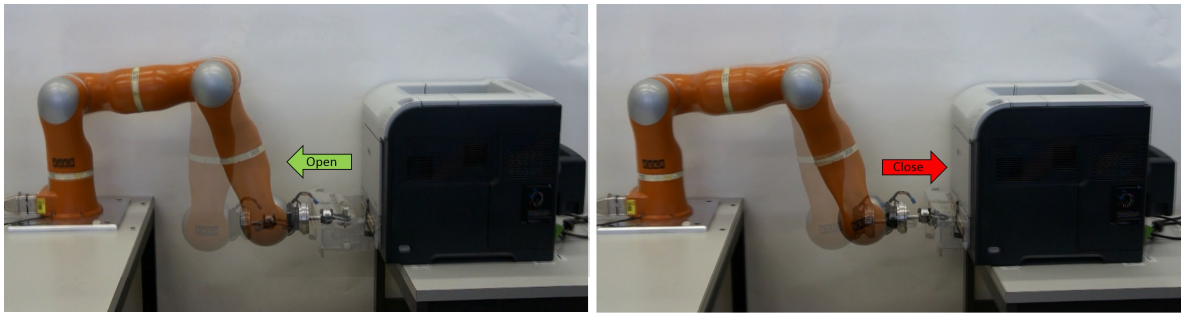


Figure 1.4: Adaptive controller implemented on KUKA LWR at EPFL.

1.1). With this setup, a human guides the robot to open the tray (as shown in Figure 1.3). End-effector poses and interaction forces were measured and used to estimate the parameters of a simulated friction model to test our controller (refer to Appendix A for estimated parameters and simulation results). The proposed adaptive controller was then successfully implemented on two robotic platforms: (i) the KUKA LWR arm at EPFL (Figure 1.4) and the (ii) HRP-4 Humanoid robot in collaboration with the CNRS team (Figure 1.7).

For the KUKA LWR experiment we tested the adaptive controller on different tasks. The first experiment involved opening and closing a printer's tray, which presented an abrupt transition between pre-sliding and sliding behavior of an empty tray and one with 2.5kg of paper. In Figure 1.5, we show the results of the latter experiment. The first row shows the tray's position, and it is easily seen that around second 3, the friction transition occurs and the movement suddenly starts. In the second row, the feed-forward force is plotted. As expected it increases up to some point where it exerts enough force so that the transition from elastic to plastic displacement happens and movement starts. The next rows show the adaptive controller parameters. For the closing part of the experiment, it is easier to start the movement, but at the end it is necessary to exert a higher force to push the tray back into the printer.

The second experiment involved opening/closing a drawer with different loads (5kg difference). In Figure 1.6 we show plots for one of these experiments. As can be seen, our proposed controller performed as desired for both of these tasks, thus showcasing the capability of adapting to unseen varying task conditions. For a detailed description of the results and estimated parameters for the other experiments please refer to Appendix A.

1.2 Implementation on the HRP 4 Robot

We explain in this section how the printer trailer opening was implemented with the Stack-of-Tasks (SoT) control framework on the HRP-4 humanoid robot and integrating the adaptive controller.

For this experiment, vision is not integrated as the main concern is to assess the integration of the adaptive controller to the SoT and the capability of the humanoid robot HRP-4 to achieve the task. Also the robot is put in front of the printer and we assume the position and orientation of the printer w.r.t the robot is perfectly known. Also, the SoT is combined with a finite state machine (that will be replaced by CRAM in Y4) to progress from a step to the other in the printer trailer opening and closing process.

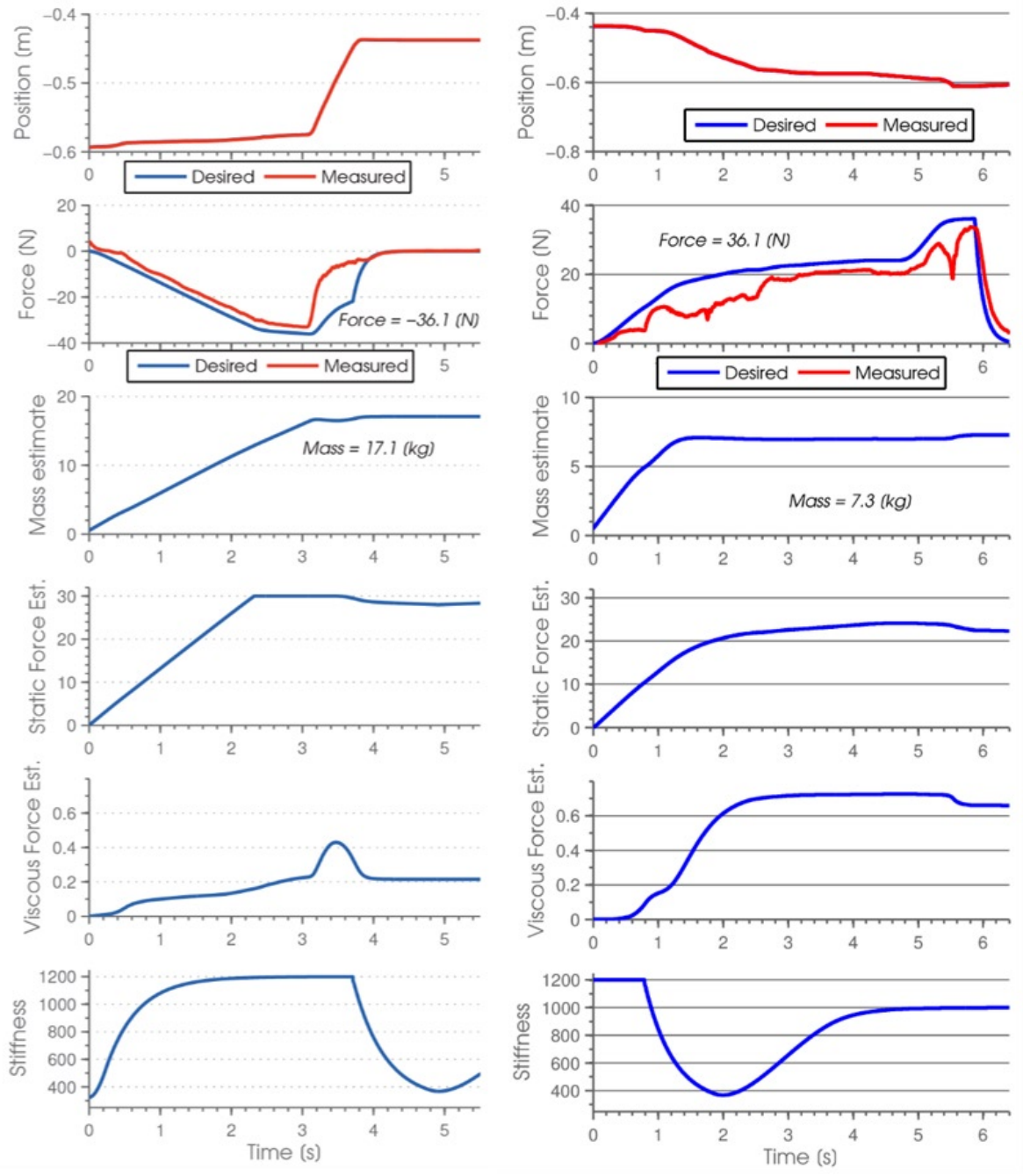


Figure 1.5: Plots for parameter estimation using the proposed adaptive force controller for (left) opening a printer's tray with 2.5kg of paper and (right) closing the same tray.

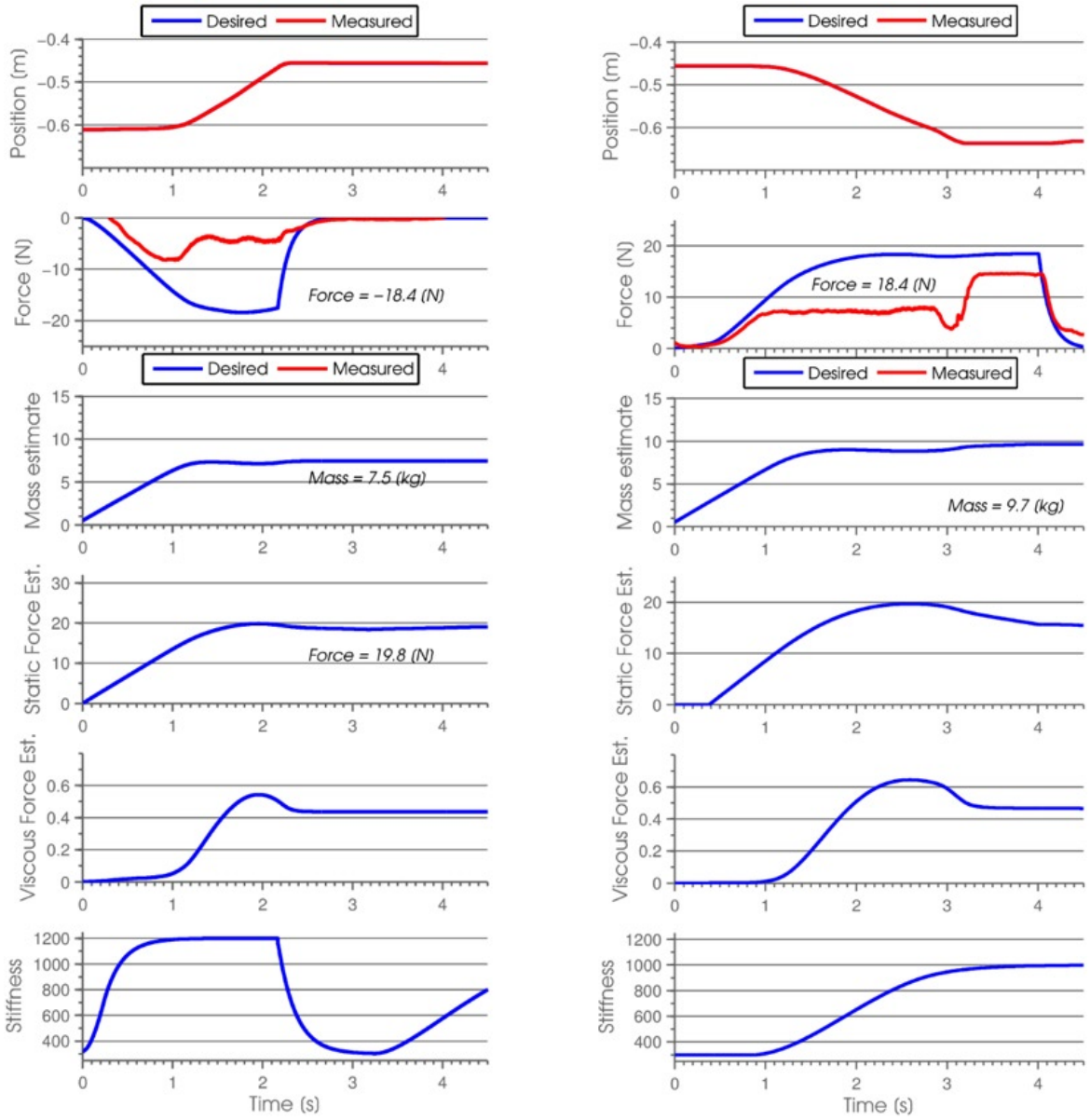


Figure 1.6: Plots for parameter estimation using the proposed adaptive force controller for (left) opening a drawer with 5kg of mass inside and (right) closing the same drawer.

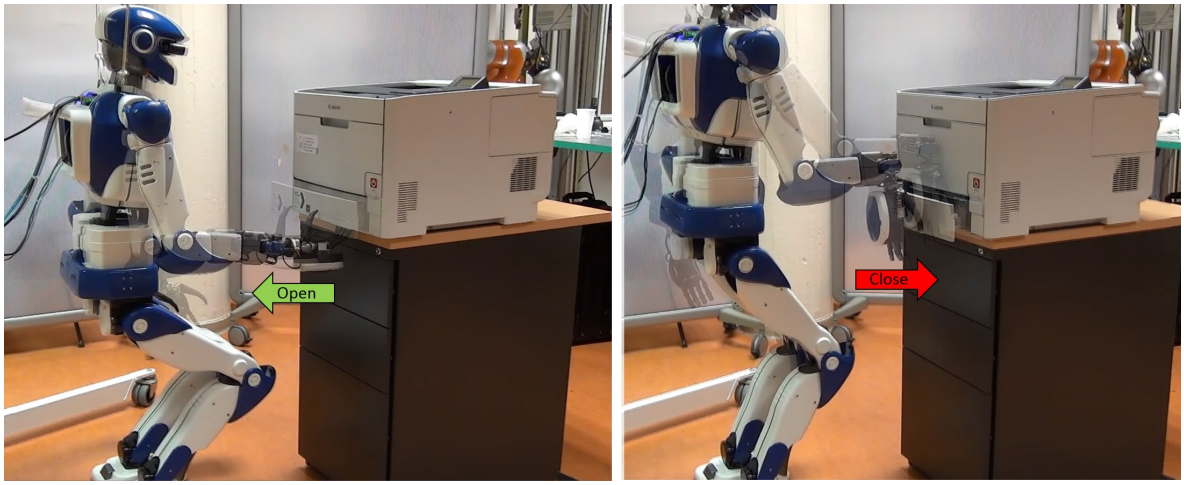


Figure 1.7: Adaptive controller implemented on HRP-4 Humanoid Robot at CNRS.

The following steps where made for the experiment (excerpts from the Python script):

```
def step():
global stateMachineStep
if stateMachineStep == 0:
   crouch()
```

In this step the SoT contains only the walking task that are in the lexicographical order as follows: CoM(x,y) $\succ$ LeftFoot(6d position) = RightFoot(6d position) $\succ$ Waist_Height(h) $\succ$ Robot_Task_Position(fixes the chest and head joint angles at a constant value). In this last task we simply set the height of the HRP-4 waist to be at the level of the printer so that the motion of the arm can be made simpler.

```
elif stateMachineStep == 1:
      initialPosition()
```

Here we add to the current SoT the following two tasks at lower priority and in order:
Right_Wrist (position 6D) = Robot_RightFingers(value)

```
elif stateMachineStep == 2:
      initialPosition()
elif stateMachineStep == 3:
      moveInPrinter()
```

In this step the SoT is not changed. Only the position of the arm is modified so that it goes near the trailer and then change the height ($z-$axis) so that the fingers are positioned inside the trailer handle in order to be ready to be pulled.

```
elif stateMachineStep == 4:
     robot.features[...].errorIN.value = (1.5708,0.65,)
```

Closes the hands's fingers. The goal is to ensure that the gripper does not pull out the trailer handle when the pulling starts.

```
elif stateMachineStep == 5:
      startImpendance()
```

At this stage, the SoT has the following tasks and their prioritized order:
CoM(x,y) $\succ$ LeftFoot(6d position) = RightFoot(6d position) $\succ$ Waist_Height(h) $\succ$ Robot_Task_Position(values) $\succ$ taskRHOrient(orientation matrix): this task fixes the orientation of the gripper during motion $\succ$ taskRHz() which constraints the motion to be constraints on a plan (yz-plan) $\succ$ taskRH: which is the task defining the positioning of the gripper (i.e. its motion) and hence the opening of the trailer. This is the task which relates to the Impedance entity that is defined by the adaptive controller.

The remaining `stateMachineStep == 6, 7, 8` are only waypoints that define values of the gripper to release the gripper from the printer trailer's handle and position it so as to close back the trailer. Therefore, we will not detail them.

```
elif stateMachineStep == 9:
      startImpendance2()
```

At this stage, the SoT has the following tasks and their prioritized order:
CoM(x,y) $\succ$ LeftFoot(6d position) = RightFoot(6d position) $\succ$ Waist_Height(h) $\succ$ Robot_Task_Position(values) $\succ$ taskRHOrient(orientation matrix) $\succ$ taskRH: which is the task defining the positioning of the gripper (i.e. its motion) and hence the closing of the trailer. This task also relates to the Impedance entity that is defined by the adaptive controller (for closing). Note that in this step we do not constraints the gripper on the $yz-$plan.

The remaining steps (after the closing) are ignored. Note also that the finite state machine we adopted for this experiment is sequential and the steps are triggered manually (for debugging purposes). The adaptive controller developed by EPFL is wrapped into a SoTAdaptiveCtrl plug-in that returns a desired force sent to the Impedance entity which returns the position that feed the taskRH task in the SoT. The current position of the gripper (robot hand) is computed from forward kinematics and sent in a closed-form way to the adaptive controller.

Chapter 2

Constraints extraction for sequences of tasks

In the work performed this year we have refined the approaches proposed previously for performing task segmentation and extracting the task-specific constraints. We test the performance of these methods on a new task, demonstrated kinesthetically by a human: rolling a pizza dough. The task will be shown in the demonstrator and its particularities are presented below. This work contributes to the *Task 5.3 Learning adaptive stiffness control that has the desired effects*.

We investigate the problem of performing task segmentation and extracting task-specific constraints from tasks whose manipulated objects are deformed by the task itself. Within the umbrella framework of pizza making we start by addressing pizza dough rolling from a previously prepared dough ball to a flattened semi-circular dough disc with a desired area (as seen in Figure 2.1). Thus, the need for monitoring the shape of the object is necessary in order to reach the desired effects on the object and the goal of the task.

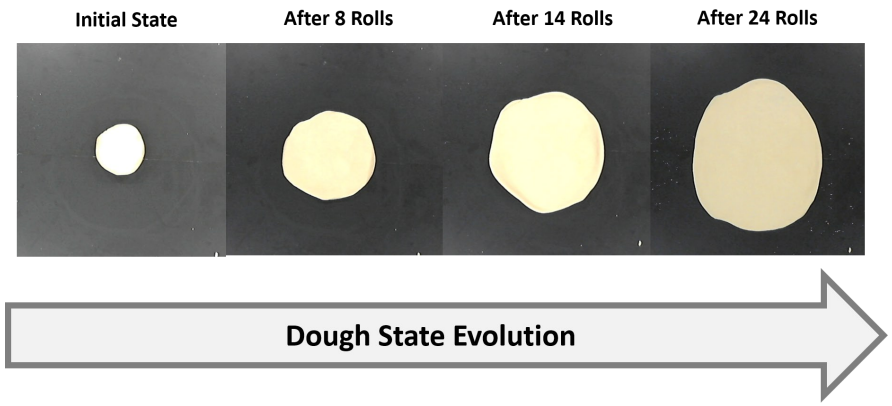


Figure 2.1: Top view of the evolution of dough state during rolling from a human demonstration.

2.1 Experimental setup and recorded demonstration data

In order to monitor the shape of the dough while recording kinesthetic demonstrations of the task, we used the experimental setup shown in Figure 2.2. We set a standard webcam on top of the robot and table (previously calibrated to the robot coordinate system), which is used to record images from the demonstration of rolling sequences in order to infer the desired effect of the sequence on the dough.

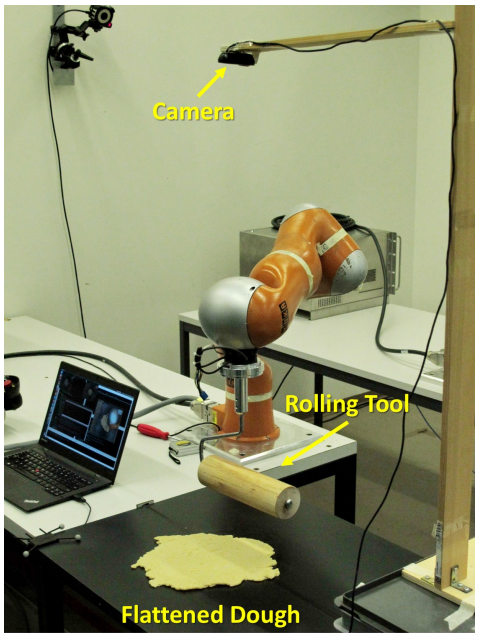


Figure 2.2: Experimental setup for recording kinesthetic demonstrations of rolling and monitoring the dough position and shape.

In Figure 2.3, plots of a recording of 3 consecutive rolling sequences are shown. We store position (x, y, z), orientation $(roll, pitch, yaw)$, forces (f_x, f_y, f_z) and torques (τ_x, τ_y, τ_z) sensed on the end-effector of the robot (i.e. the rolling tool). Following our previous work, we assume the rolling task is a sequence of primitive actions. For example, one roll sequence involves: firstly, reaching the dough at a specified position wrt. the dough to start rolling; then, roll by applying a learned force pattern in the desired direction and finally, go back to an initial/home position to decide if another roll is needed and in which direction one must roll (Fig 2.4). The execution of these decision-making metrics (i.e. roll direction, desired area/shape reached) will be described in Section 2.2.

We follow the learning/execution pipeline proposed in our previous work. To recall, a model depending on the importance of the control variables is learned from demonstrations of each phase. If position is more important than force, we consider the segment as a discrete motion modeled with an attractor-based dynamical system. If force is more important than position, we learn a force pattern (together with the attractor-based dynamical system for the motion) corresponding to the axis where this is applied. By learning a stiffness modulation profile for each phase, we can use a single Cartesian impedance controller to execute the task.

Until now, we had only dealt with tasks where the attractors for each of these phases are fixed to a non-deformable object. This task is different to our previously addressed applications as the attractors for each learned task phase and the force patterns for the actual rolling phase are dependent on the shape of the dough. In order to deal with this type of tasks, the positioning of the attractors depending on the object shape and corresponding to each phase is a new component that we have included in our framework, this will be further discussed in Section 2.2.

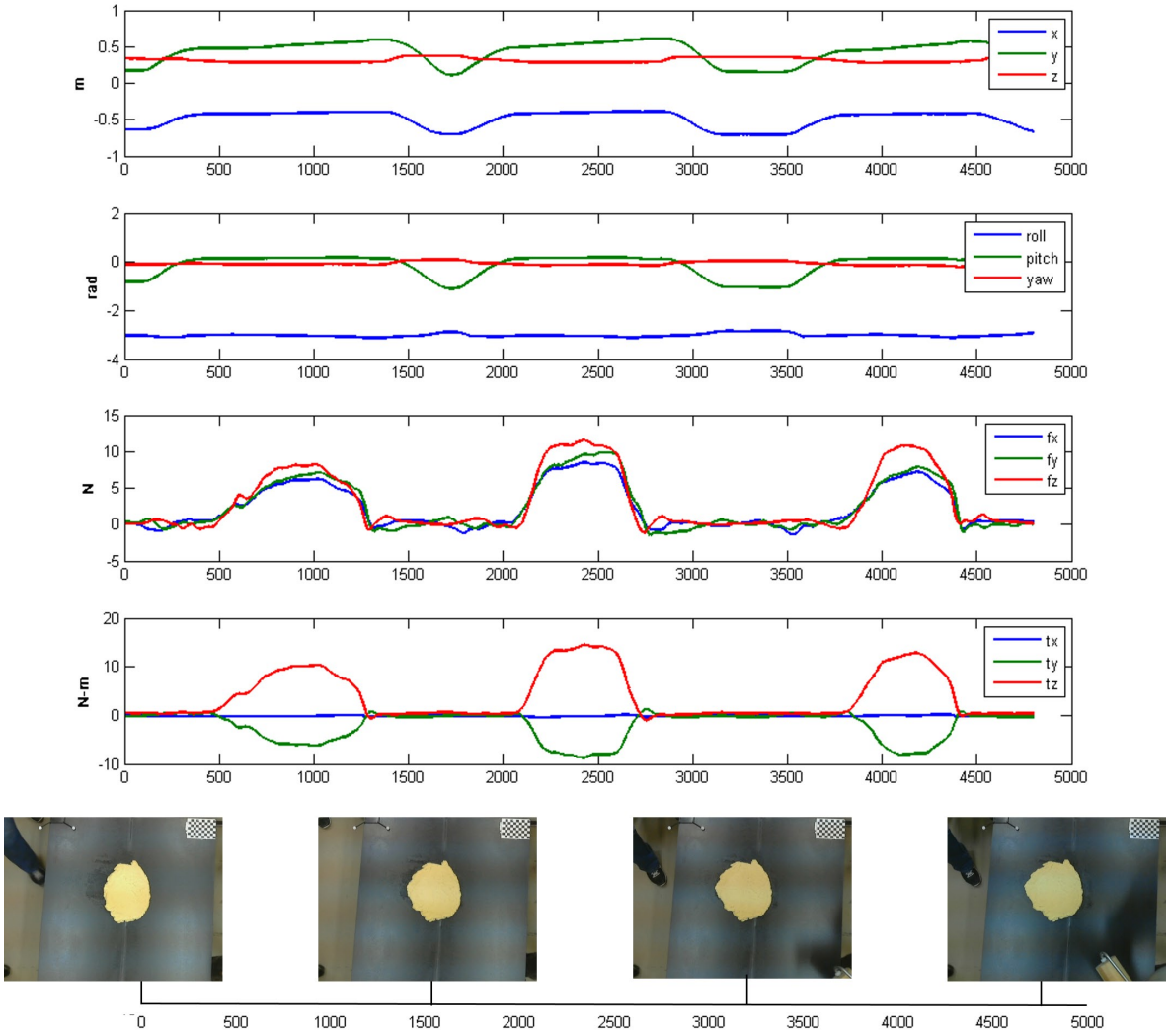


Figure 2.3: Recorded data during 3 consecutive sequences of rolling demonstrations: (1st plot) x,y,z trajectories of end-effector, (2nd plot) orientations of end-effector, (3rd plot) sensed forces on end-effector, (4th) sensed torques on end-effector and (5th) images of dough state at the beginning and end of each rolling sequence.

Moreover, we decompose our learning pipeline, by separating the segmentation and task constraint extraction components. The learning of this task begins with a probabilistic algorithm used to segment the demonstrations into primitive actions (discussed in Section 2.2) by finding similar primitive behaviors throughout multiple demonstrations in a completely unsupervised manner. Then, the extracted similar segments are fed to the constraint extraction algorithm in order to find the important control variables and learn the parameters for each action model.

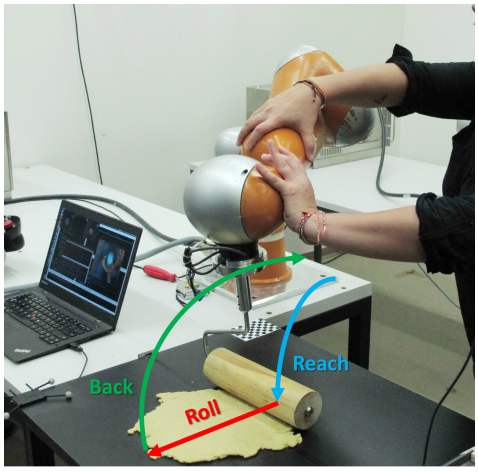


Figure 2.4: Illustration of the 3-phase rolling task (right) which includes (i) reach to dough, (ii) roll and (iii) go back.

2.2 Task Segmentation, Sequence Learning and Metric Encoding

Bayesian nonparametric approaches, allow for learning problems in sequential data to be independent of *a priori* knowledge of model parameters such as the number of hidden states, cluster or mixtures. We use an algorithm introduced by Fox et al (2009), the Beta Process Hidden Markov Model (BP-HMM), an algorithm capable of modeling multiple sequences with different switching behaviors from an unbounded set of shared primitives. Detailed evaluation and motivation for this algorithm were introduced in last year's deliverable, for an in-depth description of the BP-HMM please refer to Appendix B. We use all of the recorded variables from the demonstrations $(x, y, z, roll, pitch, yaw, f_x, f_y, f_z, \tau_x, \tau_y, \tau_z)$ to extract the desired action primitives from multiple demonstrations of the task. In Figure 2.5, the extracted primitives and segmentations points are shown for 3 demonstrations, each containing 3 consecutive rolling sequences. As can be seen, the BP-HMM can capture the similar behaviors intra- and inter-demonstrations.

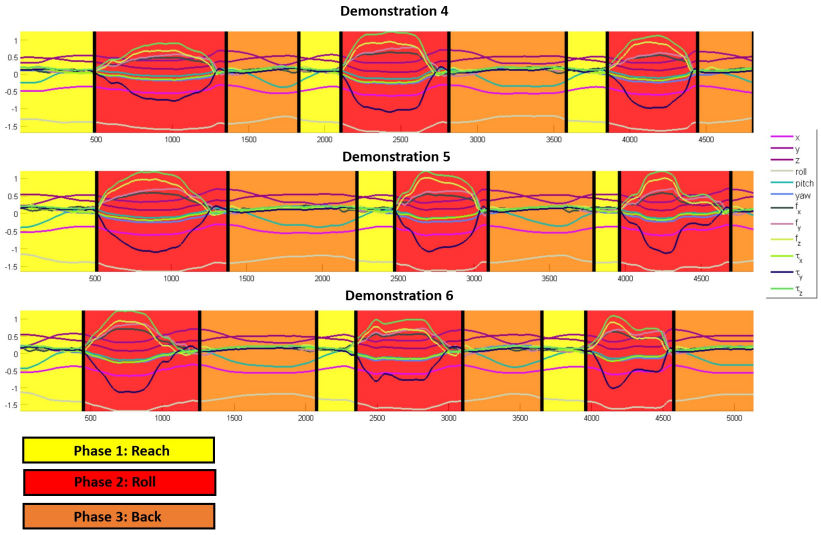


Figure 2.5: Extracted primitives and segmentation points on 3 recordings with BP-HMM.

In Figure 2.6, an illustration of the extracted segmentation can be seen for 3 consecutive rolling sequences. The BP-HMM provides an N-dimensional Gaussian distribution as the emission model $\theta_i = \{\mu_i, \Sigma_i\}$, where i is the index for each primitive and $N = 12$ for our demonstrations. In this case, we have 3 primitives, $\theta_1 : reach$, $\theta_2 : roll$, $\theta_3 : back$. In order to find the correct sequencing of the primitives, we extract the indices of the primitives for all of the segmented demonstrations. This generates the vector: $Prim\_IDx : \{1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, ...\}$. We then find the possible repeated patterns within this set of numbers. Since we know that there are 3 primitives, we take the assumption that the sequence involves these three primitives, but we do not assume their ordering. Thus, using an iterative search on $Prim\_IDx$ for a repeated pattern of length 3, we get the following possible sequences $Pattern_1 = \{1, 2, 3\}$, $Pattern_2 = \{2, 3, 1\}$ and $Pattern_3 = \{3, 1, 2\}$.

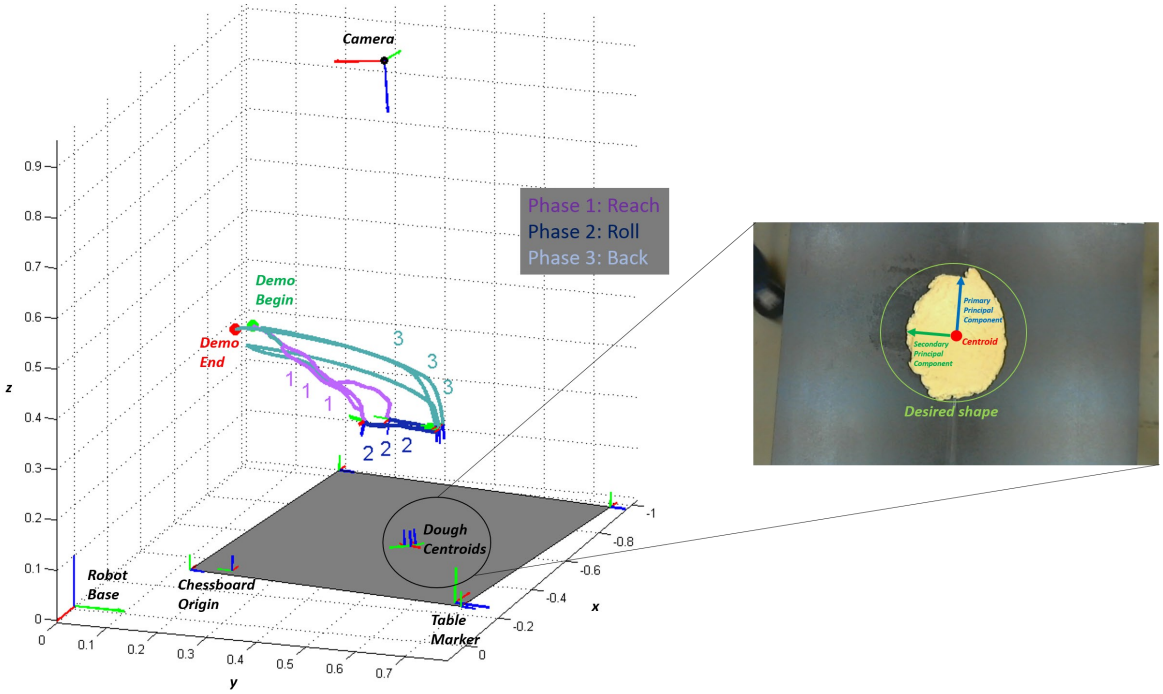


Figure 2.6: Segmented 3D trajectories of the rolling tool for 3 consecutive rolling sequences.

One of our goals is to avoid pre-defining or making assumptions on the primitives and the sequence, so we use the transition probabilities π^j learned by the BP-HMM to find the correct sequence. For each j-th time series, the BP-HMM learns a transition probability matrix for the extracted primitives which are shared throughout all time series. In order to consider all possible π^j, we compute an average transition probability matrix:

$$\bar{\pi} = \frac{1}{\kappa N} \sum_{j=1}^{N} \pi^j \tag{2.1}$$

where κ is the sticky parameter used to bias the HMM to match high self transitions and N is the total number of time-series (i.e. demonstrations) used to extract the primitives. For an explanation on how each π^j is constructed refer to Appendix B. We can then construct a stationary markov chain of the primitives using $\bar{\pi}$ as the transition matrix. Given that a transition probability between

unlikely states can yield to 0, by computing the joint probabilities of each possible sequence $Pattern_i$, we can infer the correct sequence by finding the $max(P(Pattern_i)|i = 1, 2, 3)$. Once the sequence is learned, we correlate the beginning and end of each sequence from the segmented demonstrations to an image of the state of the dough, as can be seen in Figure 2.3. We then process this image by using open source computer vision software, to: (i) detect the dough on the table, (ii) compute centroid and principal components of the dough contour (using Eigen Value Decomposition) and construct a reference frame for the dough wrt. the robot base and (iii) compute the area of the dough. The segments are then transformed to the dough reference frame in order to extract the task constraints.

However, before extracting the task constraints for each segment two important new factors have to be taken into account: (i) rolling direction and attractor positioning and (ii) a stopping metric for the overall task. From our multiple demonstrations, we discovered that all of the rolling trajectories followed the same direction as the secondary principal component of the dough shape. Furthermore, the ending of the *reaching* segments (beginning of the *rolling* segments) were clustered in beginning of that direction and the ending of the *rolling* segments (beginning of the *back* segments) were clustered on the same direction but on the opposite side (see Figure 2.7). Thus, we use the principal component analysis as the main feature to decide the direction and position of attractors for the task.

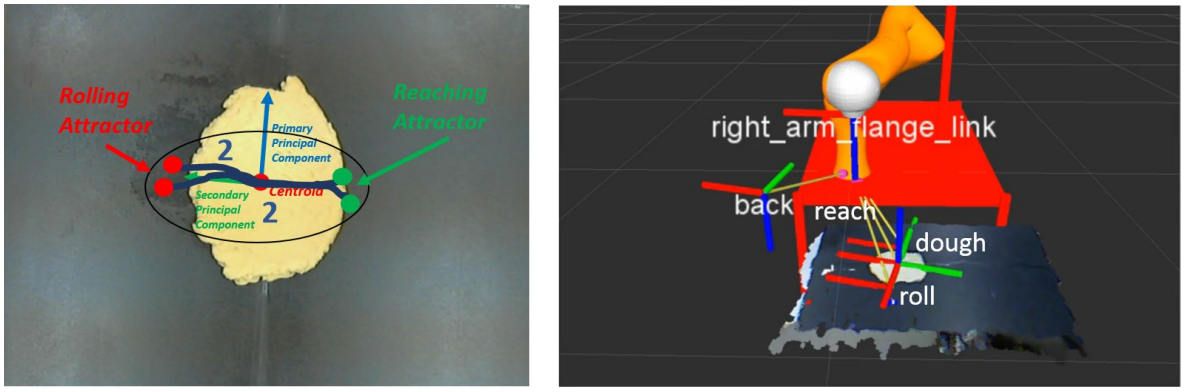


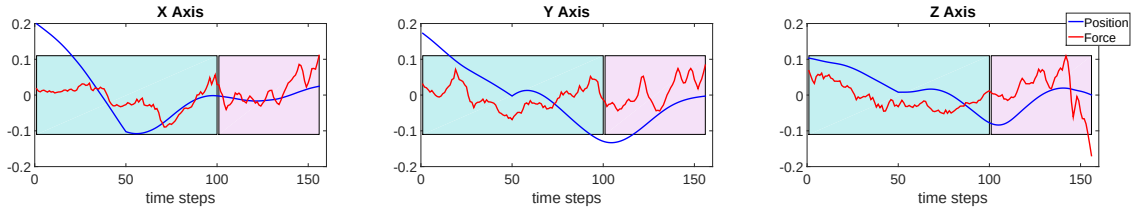
Figure 2.7: Rolling direction correlated to secondary principal component of dough shape and attractors positioned at the negative and positive directions of the respective components. On the (left) starting and ending of the segmented rolling trajectories depicted by green and red dots respectively and on the (right) the position of the attractors in 3D for execution.

For the stopping metric, we decided to use the maximum area reached throughout all demonstrations, as we assume that the demonstrations were optimal and stopped because the overall goal of the task was reached.

2.3 Extraction of task constraints

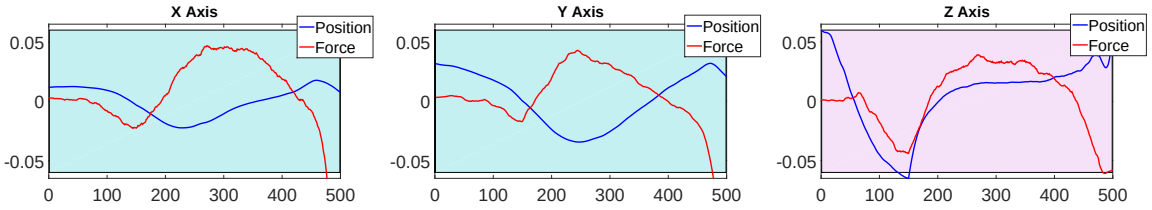
For each of the segments determined above we analyze the task constraints using the approach proposed in our previous work Pais et al (2014). For this task we focus only on the extraction of the variables of interest for each axis and each segment. The extraction of reference frame is not necessary as only one object was used throughout the task (i.e. the dough).

We analyzed the important variables on the "reaching" segment (see Fig. 2.8), and determined that this is a position-controlled segment. However the reaching motion ends when the end effector is in contact with the environment, therefore the last part of the segment has force as a variable of interest. In this case the stopping condition does not require only reaching an attractor within a given threshold, but also force information. The Stiffness modulation factor λ corresponding to this segment is shown in Fig. 2.12 (a).



(a) (b) (c)

Figure 2.8: Criteria for the "reaching" segment in the dough rolling task. Blue background highlights a position controlled part and red background highlights force a controlled part of the motion.

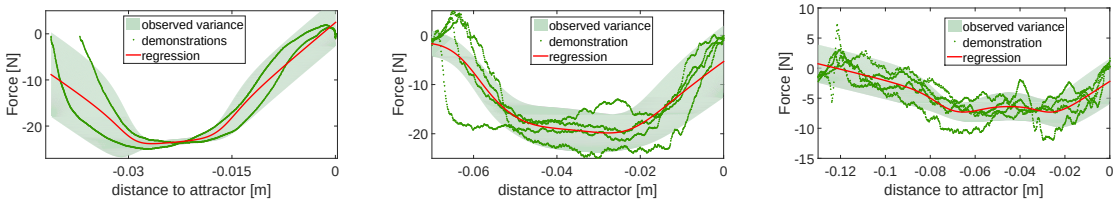


(a) (b) (c)

Figure 2.9: Criteria computed for measurements of end effector force and position in the reference frame of the dough. The criteria was computed for the data corresponding to the "rolling" segment, obtained previously. The important variables extracted as task constraints are the position for the X and Y axis and the force for the Z axis, perpendicular to the dough.

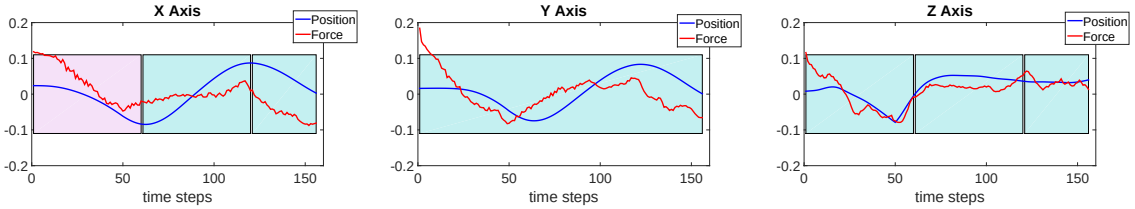
For the "rolling" task segment the task variables were analyzed, as seen in Fig. 2.9, revealing force as an important variable on the Z axis, while position control should be performed on the other two dimensions. The rolling action is demonstrated several times, as a full task demonstration presumes rolling from a spherical dough to a complete flat dough. Throughout the rolling iterations, the task constraints remain the same, however the force profile changes, as the surface of the dough increases. Therefore we encode 3 separate models for different phases of rolling, see Fig. 2.10. The profile in Fig 2.10(a), corresponds to a dough area of up to $0.012\ m^2$, in (b) the dough area is up to $0.018\ m^2$, and in (c) of up to $0.045\ m^2$. The corresponding stiffness modulation during rolling is shown in Fig. 2.12 (b).

The last action in the task consists of moving away from the dough. The criterion for determining the important variables is shown in Fig. 2.11. This is a position-weighted controller, however the action starts while the end effector is still in contact with the dough, therefore in the first part of the task the force criterion is larger. The corresponding stiffness profile is shown in Fig. 2.12 (c).



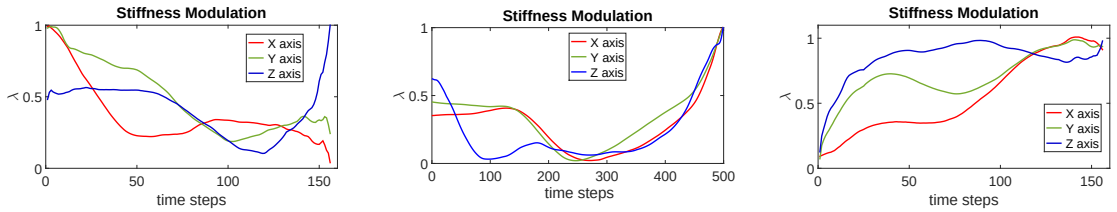
(a) (b) (c)

Figure 2.10: The profile of the applied force changes through iterations as the dough size increases. We have identified three distinct phases for which we use the force profiles above.



(a) (b) (c)

Figure 2.11: Criteria for the "moving away from the dough" segment. Blue background highlights a position controlled part and red background highlights force a controlled part of the motion.



(a) reaching (b) rolling (c) moving away

Figure 2.12: Stiffness modulation used in the rolling dough task: (a) in the "reaching" segment; (b) in the "rolling" segment; (c) in the "moving away" segment

Each segment of the task, with it's corresponding constraints is described by the format proposed in Table 2.1. This information is stored in KnowRob. We use this representation for storing additional semantic information about the learned models, that otherwise is implicit. An example of such a specification is provided in Table 2.3.

The full task specification, comprising the list and ordering of the segments as observed in the learning phase are later described using the specifications in Table 2.2.

Constraints	Description
id	the segment id in the current task sequence
object	the object of interest
attractor	the relative positioning with respect to the object of interest
threshold	reaching threshold used for stopping the motion
model	for each of the stored models we specify:
model type	CDS for reaching motions; GMR for force encodings etc.
model file	name of the file
dt	the sampling rate at which the model should be queried
gmm type	master/slave/coupling for CDS models
input type	the variables used as input to the model (position/velocity/force etc)
input dim	the dimensions used for input (X, Y, Z)
output type	output type of the model (position/velocity/force etc)
output dim	the dimensions used for output (X, Y, Z)

Table 2.1: **Segment–level specification.** Contains the list of constraints for the current segment, extracted from learning. These are encoded as *.yaml files and stored in Knowrob. Generally the specification comprises the object of interest and attractor (which represent the reference frame to be used in that segment), and the important variables in that segment (position, and force, stiffness) and their corresponding models. For controlling the next desired pose in the current "dough rolling task" we use CDS (coupled dynamical systems, Shukla and Billard (2012)).

Task	Description
sequence	the learned sequence from the segment ids (i.e. $Pattern_1 = \{1, 2, 3\}$)
name	the object/objects of interest
sequence_direction	directionality of the task (i.e. 2nd principal component of dough)
phase	specifies the current phase of the task
stop	the stop condition for the task (i.e. desired shape/area reached)

Table 2.2: **Task–level specification.** It contains the description of the task sequence obtained from learning. The segments are listed as they were observed during the demonstration. Additionally other high-level parameters are specified, such as: the stop condition, the current task phase, and the direction for the next rolling action.

Constraints	Description
id:	2.0
object:	dough
attractor:	- [-1.0, 0.0, 0.0, 0.10]
	- [0.0, 1.0, 0.0, -0.10]
	- [0.0, 0.0, -1.0, 0.24]
	- [0.0, 0.0, 0.0, 1.00]
threshold:	0.002
- modeltype:	CDS
modelfile:	
- name:	masterGMM.txt
dt:	0.02
gmmtype:	Master
input:	type: Position
	dim: [x, y, z]
output:	type: Velocity
	dim: [x, y, z]
- name:	slaveGMM.txt
dt:	0.02
gmmtype:	Slave
input:	type: Orientation
	dim: [x, y, z]
output:	type: Velocity
	dim: [x, y, z]
- name:	cplGMM.txt
dt:	0.02
gmmtype:	Coupling
input:	type: Other
	dim: ['*', '*']
output:	type: Other
	dim: ['*']
- modeltype:	GMR
modelfile:	
- name:	stiffnessGMM.txt
dt:	0.02
input:	type: Position Error
	dim: $[\widehat{x}, \widehat{y}, \widehat{z}]$
output:	type: Stiffness
	dim: [x, y, z]
- modeltype:	GMR
modelfile:	
- name:	forceGMM.txt
dt:	0.02
input:	type: Position Error
	dim: $[\widehat{x}]$
output:	type: Force
	dim: [z]

Table 2.3: Example specifying the "rolling" segment in the "dough rolling" task.

Chapter 3

Learning object-level impedance control

Our goal in this part of work is to learn the object-level impedance control for robust grasping under perturbation. First, the desired initial impedance characteristics for a given robotic grasp is extracted from the human demonstration in the object's frame. Then, starting from this extracted grasp impedance, we propose an approach that can properly adapt the grasp impedance in order to keep the grasp stable under external perturbation. A brief summary of these two parts of work are given in the following section.

Relative impedance for robust grasping In this part of work, we want to extract the grasp stiffness from human demonstration. During the demonstration, an object is grasped by a human demonstrator with eyes closed. The grasped object is perturbed by another person randomly and the displacement of the object is recorded $\{\mathbf{x}^i, i = 1...N\}$.

Our idea behind the stiffness extraction is quite intuitive: the object stiffness in one direction is inversely proportional to the variance of displacement under perturbation in the corresponding direction. From this assumption, we can learn the relative stiffness for robust grasping in different directions from recorded data as follows:

$$K = \alpha\{\frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}^i - \mathbf{x}_r)(\mathbf{x}^i - \mathbf{x}_r)^T\}^{-1} \tag{3.1}$$

where $\alpha \in \mathbb{R}^+$ is a ratio parameter that needs to be set manually and $\mathbf{x}_r \in \mathbb{R}^6$ is the object initial (and desired) position and orientation.

An example for the demonstration of human expert and the implementation on Allegro hand is shown as Fig. 3.1. The experiments on robotic hand demonstrate the effectiveness of our approach. For more details, one can refer to our ICRA paper (Li et al, 2014b).

Learning of grasp adaptation In this part of work, we extend the object level impedance approach in (Li et al, 2014b) to vary the grasp impedance dynamically in order to keep the grasp stable. To this end, we propose an adaptation scheme for the object-level impedance controller using tactile feedback S to change the control parameters K(stiffness) and L(rest length) so as to maintain a stable grasp. Since the objective of grasp adaptation is to ensure the stability of

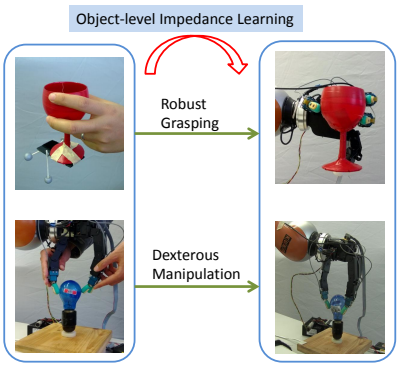


Figure 3.1: The object-level impedance for robust grasping. The **left** figures are showing the human demonstration, while the **right** figure are the implementations on Allegro hand.

the final grasps, a grasp stability estimator will be designed to dynamically predict the stability of the current grasp.

Once a grasp is predicted as unstable, the corrective actions are launched and driven by the current tactile information. In our controller, grasp adaptation consists of changing the object level impedance controllers according to the tactile feedback. Specifically, we will adapt the grasp stiffness K and the rest length L. The intuitive idea of grasp adaptation is to find a similar stable grasp in the trust grasp density region using our learned grasp stability model. The general framework for grasp adaptation is shown in Fig. 3.2. The two main components of this framework, i.e., stability estimation and grasp adaptation, are highlighted and for more details of this work, one can refer to our IROS paper (Li et al, 2014a).

Hierarchical fingertip space for adaptive grasping In our recent work, we extend the work of grasp adaptation by integrating the grasp planning and the grasp execution. In our previous work of grasp adaptation, our system relocates one fingertip using local exploration without any prior knowledge about the object and the feasible grasps that the hand can render. By using the concept of Hierarchical Fingertip Space, the feasible grasps and the possible fingertip relocations are first planned for a given object. Then during the grasp adaptation, the planned grasps are reused to make more intelligent local exploration in order to stabilize the grasp.

To be more specific, during the grasp planning, our system generates grasps in the Hierarchical Fingertip Space by optimizing both the grasp stability and the adaptability. Once a grasp is executed, our system feeds tactile readings and grasp information back to a probabilistic model learned over both tactile readings and hand configurations, and estimates its stability status online. When a grasp is predicted to be unstable, our system attempts to stabilizes the grasp by either adapting the grasp stiffness or relocating fingertips, which is planned in the Hierarchical Fingertip Space by fast graph exploration and pruning. The system is implemented on an Allegro hand mounted on a Kuka LWR arm, we present a broad set of experimental evaluations and show that this system improves the robustness of fingertip grasping, and that it achieves the performance that can not be gained by either subsystems individually. More details about this work is reported in the Appendix E, which is a joint work with KTH(WP4).

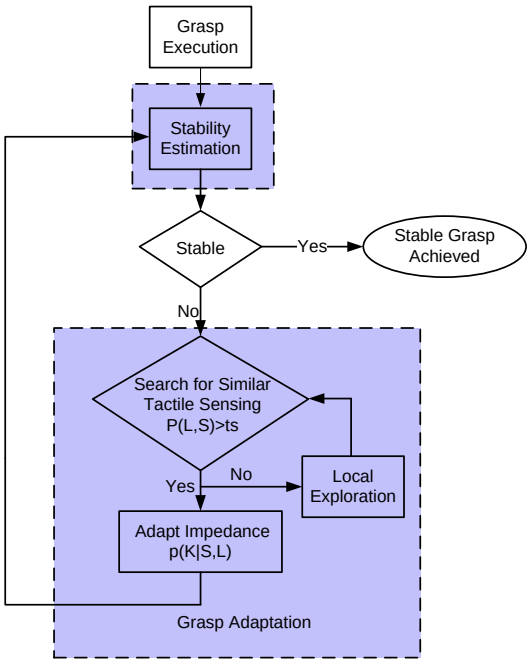


Figure 3.2: The pipeline for grasp adaptation using tactile sensing, including two main components: *Grasp Stability Estimation* and *Grasp Adaptation*.

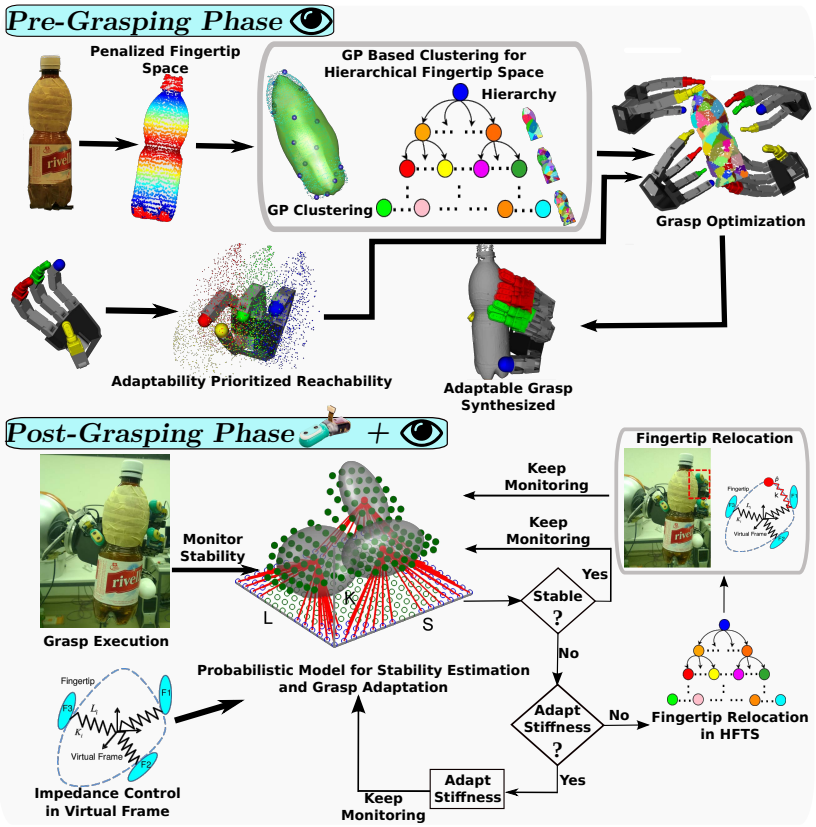


Figure 3.3: Adaptable fingertip grasping in the *Hierarchical Fingertip Space*.

Chapter 4

Appendix A

[Full Text] Ponton, B. and Billard, A. (2014) *Impedance Controller for Opening a Printer's Tray Part 2.* École Polytechnique Fédérale de Lausanne (EPFL)

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Impedance Controller for opening a Printer's Tray

Report

September 19, 2014

Supervised by

Prof. Dr. Aude Billard
Lucia Pais, Nadia Barbara
Klas Kronander, Joel Rey

Author

Brahayam Ponton

1 Printer's Tray Opening Task

1.1 Introduction

Robots which are capable of safely interacting with humans and operating in uncertain and dynamic environments is an essential factor for bringing them one step closer to a future where they can be an active part of our daily lives. It is clear that humans outperform robot capabilities in tasks requiring contacts, such as locomotion or manipulation, and there is strong evidence that the key behind this is compliance. Therefore, from the robot's perspective humans constitute a great model from where to imitate. In this work, we study the task of opening a printer's tray. In particular, we are interested in extracting a control principle that allows to successfully perform the task while dealing during task execution with the transition between static and dynamic friction inherent to the task. This principle could also be applied for similar tasks like opening a bottle, adjusting a light bulb, opening or closing a drawer, among others. Our approach to the problem is to adaptively build a model of the friction to be used for feed-forward compensation and to control the robot's compliance to compensate for not modeled dynamics. Such a system is capable of adapting its behavior for varying task conditions, shows a compliant behavior but is still able to handle external perturbations, resulting in a robust control structure. In the following, we will review related work.

There is extensive literature regarding friction modeling and control, initially developed for precision control of servo mechanisms and more recently motivated by systems requiring micro-precision control, haptic human-machine interfaces, among others. Friction compensation can be divided into two groups. On the one hand, there exists non-model based techniques for friction compensation such as high-gain feedback control [3], where sufficiently high gains make the effect of other parameters variations negligible, for a robot this means for practical matters no compliance; dither [2] is another example, where friction between two objects is reduced by subjecting one of the objects to micro-scale vibration. This is similar to strategies used by humans and to be taken into account as one of the behaviors of a rich repertoire.

On the other side, there are model-based compensation techniques that try to cancel friction force effects by feed-forward compensation. They try to model the underlying physics and their success directly depends on the quality of the model, being understood as its capability to reproduce the observed friction behavior. One of the simplest ways of model-based friction compensation is the use of a parametrized policy, whose basis functions are experimental laws that represent friction at different velocity conditions.

The richer the basis functions, the better friction can be approximated and modeled. The main drawback of this approach is that it cannot explain predominant effects at low velocities, such as Stribeck effect, pre-sliding displacement or memory-dependent behaviors. To overcome these issues, modern approaches model friction as a single-state or multi-state system of differential equations.

One of the first attempts to model friction using differential equations was undertaken by Dahl. His goal was to be able to describe with his model the pre-sliding behavior. For that he introduced the notion of modeling an internal state of the friction, now known as the average deflection of the bristles, that tries to describe pre-sliding behavior seen in friction as a stress over the quantum bonds of the contact surfaces. A further improvement to the Dahl model was proposed in the LuGre model, which can describe static and dynamic phenomena. As a dynamic model, it can model memory-dependent behaviors, it captures the Stribeck effect and can describe stick-slip motion [6]. Although it can capture more friction effects, there are still discrepancies, such as that it exhibits a drift phenomenon, not present in the world. Several modifications have been considered to account for this problem. One of the most important results is the Elasto-Plastic friction model. It has clearly identified that both the Dahl and LuGre models always include a plastic (irreversible) component when modeling the pre-sliding displacement. By noting this inaccuracy, this model can significantly improve its quality by modeling pre-sliding displacement in three different phases, namely, elastic displacement (reversible), mixed elastic-plastic displacement and plastic displacement or sliding [4, 5]. An example of multi-state model is the Generalized Maxwell-slip model [1]. In this model, friction is modeled as a group of simple systems (slip-blocks and springs with different properties) connected in parallel. The total friction comes from the sum of the simple systems. It can reproduce more complex behaviors including hysteresis with no-local memory present in the pre-sliding regime and frictional memory in sliding regime.

For many applications, such as precision control of servomechanisms, it suffices to perform once identification of the friction parameters of the motor to be able to use the friction model for compensation. In our case, we are interested in extracting a control principle able to cope with task-dependent varying friction conditions. Therefore, performing parameter identification for each task is not desired. What we would like to have is an adaptive controller, whose friction estimation can approximate if possible all friction conditions. In this work, we study different alternative adaptive controllers for friction compensation and complement them with a compliance controller, with the goal of maximizing robustness in task execution and re-usability for different tasks and conditions.

1.2 Summary of the report

This report will summarize the progress in the controller for the task of opening a printer's tray. Improvements and ongoing work will be presented taking as starting point the controller developed earlier and explained in detail in the last report. First of all, a nonlinear damping term is introduced, which improves the controller's convergence. Second, the usual controller architecture is slightly modified, in order to separately perform model identification based on desired quantities (such as desired position, velocity and acceleration trajectories) that will provide nominal performance, and the feedback gains are selected so as to guarantee robust stability, based on the degree of uncertainty in the model. Finally, for comparison purposes, two different controllers are presented, one based on online reinforcement learning using an Actor-Critic method; and the other one based on an Extended Kalman Filter, where the unknown friction and inertial effects are modeled as a n-th order random walk.

1.3 Use of a nonlinear damping term

This section will present the use of a nonlinear damping term in order to improve the controller's convergence. We will analyze the simplified 1-dimensional system, we have been working with. The dynamics of the 1D system can be written as given by equation 1.1a, where $F_{fr}(\dot{x})$ represents the friction force, m represents the mass of the load and manipulator, and u is the control input.

$$m\ddot{x} = u - F_{fr}(\dot{x}) \tag{1.1a}$$
$$u = \hat{m}\ddot{x}_d + \hat{F}_{fr}(\dot{x}) - Ke - D\dot{e} \tag{1.1b}$$
$$u = \hat{m}\ddot{x}_d + \hat{F}_{fr}(\dot{x}) - Ke - K_2\tilde{E}\dot{e} \tag{1.1c}$$
$$m\ddot{x} = \hat{m}\ddot{x}_d + \hat{F}_{fr}(\dot{x}) - Ke - K_2\tilde{E}\dot{e} - F_{fr}(\dot{x}) \tag{1.1d}$$

Equation 1.1b shows the usual control input that compensates for inertial and friction effects based on building a model of the mass $\hat{m}$ and the friction $\hat{F}_{fr}$, and that compensates for unknown disturbances with a spring K - damper D system. Equation 1.1c shows the modified control law that makes use of a nonlinear damping term given by $K_2\tilde{E}$. K_2 is a gain parameter and $\tilde{E}$ is the energy difference between the current state and a desired state. $\tilde{E}$ will be formally defined later; the most important part here is to realize that this term will either take out or introduce energy into the system depending on the energy difference $\tilde{E}$, which will drive the system towards the desired energy level. First, we will remember some assumptions and notation. For simplicity, we will use a static model to compensate the friction $F_{fr}(\dot{x})$. This means that, we assume that friction can be represented as a parametrized policy, with basis functions $\Psi(\dot{x})$ and parameter

vector θ, as given by equation 1.2a. The results developed in this section can be easily extended to other types of friction compensation. We compensate for friction as given by $\hat{F}_{fr}(\dot{x})$ in equation 1.2b. We also define the terms $\tilde{\theta}$ and $\tilde{m}$ as the errors between the true values and the estimated values of the friction parameters and the mass of the system, as shown in equations 1.2d and 1.2e respectively. Its time derivatives are shown in equations 1.2f and 1.2g respectively.

$$F_{fr}(\dot{x}) = \Psi^T \theta \tag{1.2a}$$
$$\hat{F}_{fr}(\dot{x}) = \Psi^T \hat{\theta} \tag{1.2b}$$
$$\Psi_s(\dot{x}) = \left[\text{sign}(\dot{x}), \dot{x}, \dot{x}|\dot{x}|, \sqrt{|\dot{x}|}\,\text{sign}(\dot{x})\right]^T \tag{1.2c}$$
$$\tilde{\theta} = \theta - \hat{\theta} \tag{1.2d}$$
$$\tilde{m} = m - \hat{m} \tag{1.2e}$$
$$\dot{\tilde{\theta}} = -\dot{\hat{\theta}} \tag{1.2f}$$
$$\dot{\tilde{m}} = -\dot{\hat{m}} \tag{1.2g}$$

Additionally, we define position, velocity and acceleration error terms (e, $\dot{e}$, $\ddot{e}$), as follows:

$$e = x - x_d \tag{1.3a}$$
$$\dot{e} = \dot{x} - \dot{x_d} \tag{1.3b}$$
$$\ddot{e} = \ddot{x} - \ddot{x_d} \tag{1.3c}$$

where x, $\dot{x}$ and $\ddot{x}$ are current values of position, velocity and acceleration. x_d, $\dot{x}_d$ and $\ddot{x}_d$ represent desired values of position, velocity and acceleration. With the dynamics equation (eq. 1.1a) and the defined control law (eq. 1.1c), we can obtain an error dynamics equation, as follows:

$$m\ddot{x} = m\ddot{x_d} - \tilde{m}\ddot{x_d} + \Psi^T\hat{\theta} - \Psi^T\theta - Ke - K_2\tilde{E}\dot{e} \tag{1.4a}$$
$$m\ddot{e} = -\tilde{m}\ddot{x_d} - \Psi^T\tilde{\theta} - Ke - K_2\tilde{E}\dot{e} \tag{1.4b}$$

Now we are ready to define our Lyapunov function E. As normally done, we define our Lyapunov function as the energy in the system, containing the 4 terms: kinetic energy $\frac{1}{2}m\dot{e}^2$, potential energy $\frac{1}{2}Ke^2$, errors in mass estimation $\frac{1}{2}\frac{\tilde{m}^2}{\omega_m}$ and errors in friction estimation $\frac{1}{2}\tilde{\theta}^T\Gamma^{-1}\tilde{\theta}$. Where ω_m and Γ are gain parameters for tunning purposes (scalar and a matrix of appropriate dimensions). Equation 1.5a shows the defined Lyapunov function, which tell us the energy level in the system at the current state (e, $\dot{e}$, $\tilde{m}$, $\tilde{\theta}$).

It is clear that we would like all of the errors to go to zero, which means that we would like the desired energy level E_d to be zero (eq. 1.5b); but in general, we could drive the system towards an arbitrary energy level. At this point, we can realize the meaning of $\tilde{E}$, used in the control law for the nonlinear damping, which is defined in equation 1.5c as an energy difference. The time derivative $\dot{\tilde{E}}$, which is used for deriving control laws and proving stability of the controller is shown in equation 1.5d.

$$E = \frac{1}{2}m\dot{e}^2 + \frac{1}{2}Ke^2 + \frac{1}{2}\frac{\tilde{m}^2}{\omega_m} + \frac{1}{2}\tilde{\theta}^T\Gamma^{-1}\tilde{\theta} \tag{1.5a}$$

$$E_d = \text{desired energy level} \tag{1.5b}$$

$$\tilde{E} = E - E_d \tag{1.5c}$$

$$\dot{\tilde{E}} = \dot{E} - \dot{E}_d = m\ddot{e}\dot{e} + Ke\dot{e} + \frac{\tilde{m}\dot{\tilde{m}}}{\omega_m} + \tilde{\theta}^T\Gamma^{-1}\dot{\tilde{\theta}} \tag{1.5d}$$

By introducing the error dynamics equation (eq. 1.4b) into the time derivative of the Lyapunov function (eq. 1.5d), we obtain:

$$\dot{\tilde{E}} = \left(-\tilde{m}\ddot{x}_d - \Psi^T\tilde{\theta} - Ke - K_2\tilde{E}\dot{e}\right)\dot{e} + Ke\dot{e} + \frac{\tilde{m}\dot{\tilde{m}}}{\omega_m} + \tilde{\theta}^T\Gamma^{-1}\dot{\tilde{\theta}} \tag{1.6a}$$

$$\dot{\tilde{E}} = -\tilde{m}\ddot{x}_d\dot{e} - \Psi^T\tilde{\theta}\dot{e} - Ke\dot{e} - K_2\tilde{E}\dot{e}^2 + Ke\dot{e} + \frac{\tilde{m}\dot{\tilde{m}}}{\omega_m} + \tilde{\theta}^T\Gamma^{-1}\dot{\tilde{\theta}} \tag{1.6b}$$

$$\dot{\tilde{E}} = -\tilde{m}\left(\ddot{x}_d\dot{e} + \frac{\dot{\hat{m}}}{\omega_m}\right) - \tilde{\theta}^T(\Psi\dot{e} + \Gamma^{-1}\dot{\hat{\theta}}) - K_2\tilde{E}\dot{e}^2 \tag{1.6c}$$

Based on the time derivative of the Lyapunov function (eq. 1.6c), we can derive two control laws for the update of the parameters $\hat{m}$ and $\hat{\theta}$, by setting the terms in parenthesis to zero.

$$\dot{\hat{m}} = -\omega_m\ddot{x}_d\dot{e} \tag{1.7a}$$

$$\dot{\hat{\theta}} = -\Gamma\Psi\dot{e} \tag{1.7b}$$

With these update rules, the time derivative of the Lyapunov function (eq. 1.6c) can be simplified (eq. 1.8a), which means that the energy difference $\tilde{E}$ decays exponentially.

$$\dot{\tilde{E}} = -K_2\dot{e}^2\tilde{E} \tag{1.8a}$$

$$\tilde{E}(t) = \tilde{E}(0)\exp\left(-K_2\dot{e}^2t\right) \tag{1.8b}$$

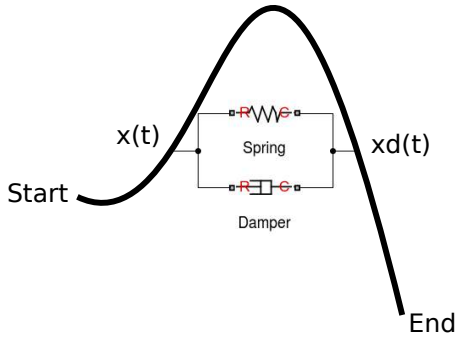


Figure 1.1: Case I: Spring-damper attached to the current and desired positions in the trajectory.

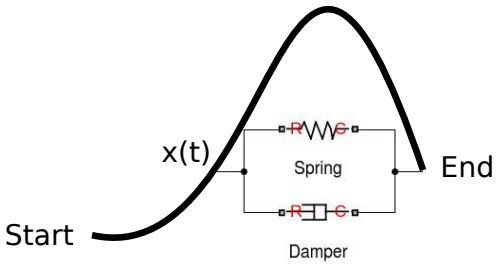


Figure 1.2: Case II: Spring-damper attached to the current and final positions in the trajectory.

This method means that the control input u has a nonlinear damping term proportional to energy difference $\tilde{E}$. But we cannot exactly know $\tilde{E}$, because it depends on the terms $\tilde{m}$ and $\tilde{\theta}$, which are the errors of estimated mass and friction parameters with respect to its true values (which we do not know). In practice, this can be solved by realizing that the parameters can be upper and lower bounded. Therefore, the errors could be upper bounded. Additionally, with high enough adaptive rates these error terms become negligible in comparison to the other energy terms.

Figures 1.1 and 1.2 show two ways of attaching the spring-damper system. In figure 1.1 the spring-damper system connects the current and desired positions, we will call this: case I. In figure 1.1 the spring-damper system connects the current and final positions, and we will call this: case II. We will present some simulation results for both cases. The goal for this task was to open the printer's tray 10cm from its initial location. For the plots with adaptive controller, the basis functions used $\Psi(\dot{x})$ are as given by equation 1.2c. The diagonal coefficients of the adaptive rate matrix for friction estimation Γ are set to 100, and the adaptive rate for inertial effects ω_m is set to 300. The natural frequency ω used to parametrize the stiffness gain K is 20. The gain for the nonlinear damping K_2 is 5.

In Figure 1.3, we can see the controller results for case I, where the spring-damper system moves along with the current and desired positions. The stiffness gain K is parametrized in the form $K = \hat{m}\omega^2$, where ω is the damped natural frequency. This means that the stiffness K is parametrized by our estimated mass and varies along with our estimation. In order to build a critically damped system, the damping gain is usually selected as $D = \hat{m}(2\xi\omega)$, where the damping coefficient ξ is chosen to be 1. In our case, we do not use the usual damping coefficient, but a damping term which is energy-dependent. The feedback term (first plot) shows the PD action composed of $Ke + K_2\tilde{E}\dot{e}$, according to equation 1.1c. At first, it increases to break static friction, but when the object starts moving, it goes below zero to make it slow down. The same

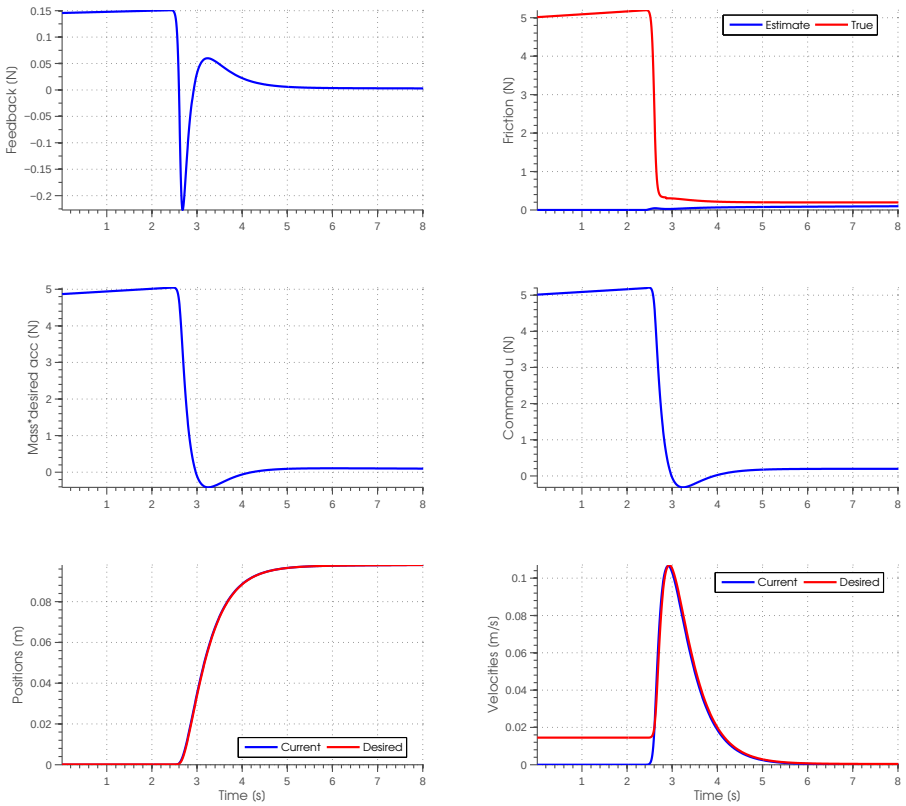


Figure 1.3: Controller for Case I.

behavior can be seen in the term for inertial compensation ($\hat{m}\ddot{x}_d$). This is closer to the behavior observed with the KUKA robot, where the demonstration shows first a high positive force and then a negative force period. Parameterizing the feedback and feed-forward terms by the estimated mass is advantageous, because the feed-forward term does most of the work to start movement, but the PD controller has the sufficient authority to regulate the behavior in presence of disturbances and perturbations. The second plot shows the friction estimation, which can be interpreted in the following way: when there is no movement, the basis functions are zero (static friction compensation) and therefore, our friction force estimate also remains at 0; once the object starts moving the controller realizes of the sudden change in acceleration due to the friction transition and will decrease the mass estimate, then both friction and mass estimates will continue increasing until convergence to its correct values (under the assumption of persistence of excitation, which is usually not true for a single short experiment (at least in simulation)).

In Figure 1.4, we can see the controller results for case II, where the spring-damper system moves along with the current position, but on the other end it is always attached to the desired final position (in our case, 10cm). The controller is the same as in case I, with the difference that we do not track a trajectory (LQR policy), but we just try to regulate the system to a final desired state. In this case, the effects of the PD controller

are presented separately. The plot named "Feedback" is the control input given by the stiffness gain times the position error, and the plot named "Energy damping term" presents the effects of the nonlinear damping term, where we can see that it is mainly taking out energy of the system during the transition time between static and dynamic friction.

Finally, Figure 1.5, shows the behavior with a normal adaptive PD controller without energy-based damping term and without friction or mass compensation. Although it can perform the task, we can clearly see that this controller that only relies on a PD, ends up with a tracking error, and in the presence of noise, it would perform worse. This highlights the importance of an architecture with feedback and feed-forward control, where the advantage is that it is possible to reduce the sensitivity of feed-forward command against perturbations and the sensitivity of feedback commands to sensor noise.

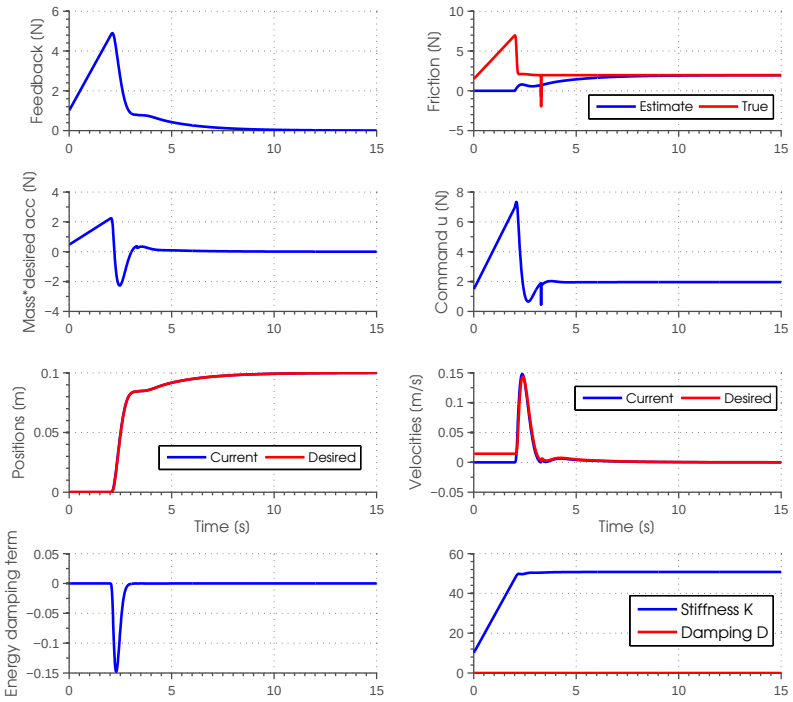


Figure 1.4: Controller for Case II.

1.4 Compensation based on desired quantities - Ongoing work

In this section, we present a controller architecture with an experimental slight modification. The idea is based on robust control methods where we have a nominal plant and an additional uncertainty term, which indicates the degree of uncertainty in our nominal model. For example, it could be a multiplicative uncertainty that indicates our uncertainty about the plant as a percentage of the nominal bode magnitude as a function of the frequency. This representation is then used to synthesize a controller

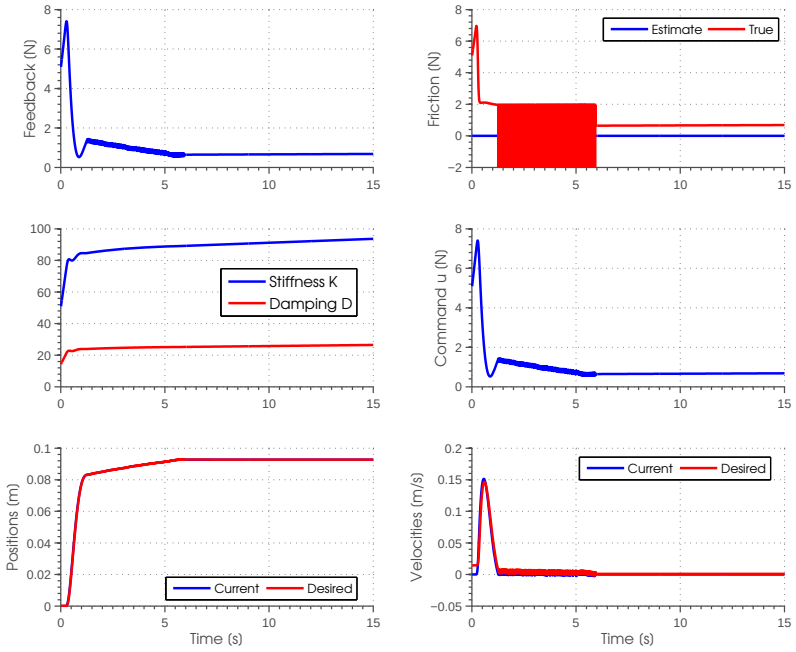


Figure 1.5: Controller for Case II, without model-based compensation, only a PD controller.

that achieves certain desired performance over all the range of possible plants, at the cost of reducing nominal performance. What we try to do here, is to perform identification of the unknown mass and friction parameters, based on known quantities, such as desired position, velocity and acceleration trajectories, which has as advantage that the regressors are noise-free quantities. Ideally, we would like then to have a measure of the confidence degree of this estimation, so that we can compensate for the rest of bounded uncertainty using a feedback controller.

In the following we develop a controller based on this idea. We first state the simplified 1D problem (eq. 1.9a) and the friction model (eq. 1.9b). We slightly modify the friction model by approximating the sign "function" with the hyperbolic tangent function, as shown in equation 1.9c. The purpose of this modification is to obtain a continuous

model, which will be useful later for deriving bounds on uncertainties. The parameter s_1 controls the slope of the hyperbolic tangent when its value is changing from -1 to 1.

$$m\ddot{x} = u - F_{fr}(\dot{x}) \tag{1.9a}$$

$$F_{fr}(\dot{x}) = B\dot{x} + \text{sign}(\dot{x})\left(F_{co} + F_{se}\exp\left(-\frac{|\dot{x}|}{C_s}\right)\right) \tag{1.9b}$$

$$F_{fr}(\dot{x}) = B\dot{x} + \tanh(s_1\dot{x})\left(F_{co} + F_{se}\exp\left(-\frac{|\dot{x}|}{C_s}\right)\right) \tag{1.9c}$$

$$m\ddot{x} = u - B\dot{x} - F_{co}\tanh(s_1\dot{x}) - F_{se}\tanh(s_1\dot{x})\exp\left(-\frac{|\dot{x}|}{C_s}\right) \tag{1.9d}$$

We use the following notation. We define position, velocity and acceleration errors (e, $\dot{e}$, $\ddot{e}$) as the difference between current (x, $\dot{x}$, $\ddot{x}$) and desired (x_d, $\dot{x}_d$, $\ddot{x}_d$) position, velocity and acceleration quantities. We also define required quantities (x_r, $\dot{x}_r$) and required errors (r, $\dot{r}$) as shown in equations 1.10f - 1.10e. $\lambda > 0$ is a free design parameter. The required quantities, for instance, the variable x_r is not only a desired velocity, but is a required velocity, meaning that if the system velocity can track the required velocity, then the position and velocity control objectives can be satisfied. Then we can define the error terms (r, $\dot{r}$) as shown in equations 1.10f-1.10g, which are equivalent by construction.

$$e = x - x_d \tag{1.10a}$$

$$\dot{e} = \dot{x} - \dot{x}_d \tag{1.10b}$$

$$\ddot{e} = \ddot{x} - \ddot{x}_d \tag{1.10c}$$

$$x_r = \dot{x}_d + \lambda(x_d - x) \tag{1.10d}$$

$$\dot{x}_r = \ddot{x}_d + \lambda(\dot{x}_d - \dot{x}) \tag{1.10e}$$

$$r = \dot{e} + \lambda e = \dot{x} - x_r \tag{1.10f}$$

$$\dot{r} = \ddot{e} + \lambda\dot{e} = \ddot{x} - \dot{x}_r \tag{1.10g}$$

Previously, we had defined the control law in the form given by equation 1.11a , where the control command u compensates for the inertial and friction effects learning the parameters $\hat{\theta}$ (eq. 1.11b) of a parametrized policy using the basis functions Ψ (eq.

1.11c). Additionally a feedback controller compensates for disturbances with stiffness K and damping term D .

$$u = \hat{\theta}^T \Psi - K \int_0^t r d\tau - Dr \tag{1.11a}$$

$$\hat{\theta}^T = \begin{pmatrix} \hat{m} & \hat{B} & \hat{F}_{co} & \hat{F}_{se} \end{pmatrix} \tag{1.11b}$$

$$\Psi^T = \begin{pmatrix} \dot{x}_r & \dot{x} & \tanh(s_1 \dot{x}) & \tanh(s_1 \dot{x}) \exp\left(-\frac{|\dot{x}|}{C_s}\right) \end{pmatrix} \tag{1.11c}$$

The parameter vector $\hat{\theta}$ was updated using the rule $\dot{\hat{\theta}} = -\Gamma \Psi r$, where the basis functions Ψ are computed using noisy measurements. Γ is a diagonal adaptation gain matrix of appropriate dimensions. Now, we propose a slightly change in the update rule and the selection of the feedback gains. For that, we proceed as follows. We introduce equation 1.10g into equation 1.9d, and obtain:

$$m(\dot{r} + \dot{x}_r) = u - B\dot{x} - F_{co} \tanh(s_1 \dot{x}) - F_{se} \tanh(s_1 \dot{x}) \exp\left(-\frac{|\dot{x}|}{C_s}\right) \tag{1.12a}$$

$$m\dot{r} = u - m\dot{x}_r - B\dot{x} - F_{co} \tanh(s_1 \dot{x}) - F_{se} \tanh(s_1 \dot{x}) \exp\left(-\frac{|\dot{x}|}{C_s}\right) \tag{1.12b}$$

Noting the following identities:

$$r = \dot{x} - x_r \tag{1.13a}$$

$$\dot{x} = r + x_r \tag{1.13b}$$

$$\dot{x} = r + \dot{x}_d - \lambda e \tag{1.13c}$$

$$\dot{x} = \dot{x}_d + r - \lambda e \tag{1.13d}$$

$$\dot{x}_r = \ddot{x}_d - \lambda \dot{e} \tag{1.14a}$$

$$\dot{x}_r = \ddot{x}_d - \lambda(r - \lambda e) \tag{1.14b}$$

we can rewrite the dynamics equation 1.12b in the following form:

$$\begin{aligned} m\dot{r} =& u - m(\ddot{x}_d - \lambda(r - \lambda e)) - B(\dot{x}_d + r - \lambda e) - F_{co} \tanh(s_1 \dot{x}_d) \\ & - F_{se} \tanh(s_1 \dot{x}_d) \exp\left(-\frac{|\dot{x}_d|}{C_s}\right) + \mathsf{Err}_1 + \mathsf{Err}_2 \end{aligned} \tag{1.15}$$

This equation can be rewritten in the following form:

$$\begin{aligned} m\dot{r} =& u - m\ddot{x}_d - B\dot{x}_d - F_{co}\tanh(s_1\dot{x}_d) - F_{se}\tanh(s_1\dot{x}_d)\exp\left(-\frac{|\dot{x}_d|}{C_s}\right) \\ &+ m\lambda(r-\lambda e) - B(r-\lambda e) + \mathsf{Err}_1 + \mathsf{Err}_2 \end{aligned} \tag{1.16}$$

where the model approximation error terms Err_1 and Err_2 are given by

$$\mathsf{Err}_1 = F_{co}\tanh(s_1\dot{x}_d) - F_{co}\tanh(s_1\dot{x}) \tag{1.17a}$$

$$\mathsf{Err}_2 = F_{se}\tanh(s_1\dot{x}_d)\exp\left(-\frac{|\dot{x}_d|}{C_s}\right) - F_{se}\tanh(s_1\dot{x})\exp\left(-\frac{|\dot{x}|}{C_s}\right) \tag{1.17b}$$

Note that in equation 1.16, we have used the identities 1.13d and 1.14b to explicitly separate the model terms containing desired quantities (which can be seen as the nominal model) and the model terms that contain error parameters (uncertainty term). In general, we could estimate a confidence interval for the certainty of the model of our nominal plant, which could be used as a bound for the uncertainties. In this case, we will just upper bound the model approximation errors and compensate them with the feedback controller. The required feedback effort could also be adaptively reduced as the certainty of our model increases.

Our next step is to bound the model errors Err_1 and Err_2, for which we use the mean value theorem. This allows us to bound the difference between the value of the function evaluated at two different points, by the derivative of the function times the difference between the two evaluated points. Then the upper bound comes just by taking the highest value of the function derivative (Remember that the hyperbolic tangent takes values between -1 and 1).

$$\begin{aligned} |\mathsf{Err}_1| &= |F_{co}\tanh(s_1\dot{x}_d) - F_{co}\tanh(s_1\dot{x})| \\ |\mathsf{Err}_1| &= |F_{co}||\tanh(s_1\dot{x}_d) - \tanh(s_1\dot{x})| \\ |\mathsf{Err}_1| &= |F_{co}||\left(\frac{d}{dt}\tanh(s_1\dot{x}_c)\right)(\dot{x}_d - \dot{x})| \\ |\mathsf{Err}_1| &= |F_{co}||\left(\mathsf{sech}(s_1\dot{x}_c)^2\right)s_1(\dot{x}_d - \dot{x})| \\ |\mathsf{Err}_1| &= |F_{co}||(1-\tanh(s_1\dot{x}_c)^2)s_1(\dot{x}_d - \dot{x})| \\ |\mathsf{Err}_1| &\leq |F_{co}||s_1(\dot{x}_d - \dot{x})| = |F_{co}s_1||\lambda e - r| \\ |\mathsf{Err}_1| &\leq |F_{co}s_1|(|\lambda e| + |r|) \end{aligned}$$

In the same way, Err_2 can be upper bounded by

$$|\text{Err}_2| \leq |F_{se}\left(s_1 + \frac{1}{C_s}\right)|(|\lambda e| + |r|)$$

The control command u can be defined in a similar way as we did in the previous case, namely, based on two components that exploit the structure of the dynamics equation 1.16: one component is a model-based or parametrized policy for compensation of friction and inertial effects u_m given by equation 1.20b, and the other component is a feedback controller u_r for robust stabilization, given by equation 1.20c.

$$u = u_m + u_r \tag{1.20a}$$

$$u_m = \hat{m}\ddot{x}_d + \hat{B}\dot{x}_d + \hat{F}_{co}\tanh(s_1\dot{x}_d) + \hat{F}_{se}\tanh(s_1\dot{x}_d)\exp\left(-\frac{|\dot{x}_d|}{C_s}\right) \tag{1.20b}$$

$$u_r = -K_d r - K_p\int r dt \tag{1.20c}$$

For readability purposes, we will rewrite u_m as a parametrized control law, with basis functions Ψ_d and parameter vector $\hat{\theta}$.

$$u_m = \hat{\theta}^T\Psi_d \tag{1.21a}$$

$$\hat{\theta}^T = [\hat{m}, \hat{B}, \hat{F}_{co}, \hat{F}_{se}] \tag{1.21b}$$

$$\Psi_d^T = \left[\ddot{x}_d, \dot{x}_d, \tanh(s_1\dot{x}_d), \tanh(s_1\dot{x}_d)\exp\left(-\frac{|\dot{x}_d|}{C_s}\right)\right] \tag{1.21c}$$

We then introduce the control law into the error dynamics equation 1.16, which becomes

$$m\dot{r} = -\tilde{\theta}^T\Psi_d - K_d r - K_p\int r dt + m\lambda(r - \lambda e) - B(r - \lambda e) + \text{Err}_1 + \text{Err}_2 \tag{1.22a}$$

$\tilde{\theta}$ is the element-wise difference between true and estimated parameters. We are now ready to define a Lyapunov function that will help us derive the adaptive control laws and provide stability guarantees. The Lyapunov function shown in equation 1.23a is similar to the functions we have been using. It contains a kinetic energy term $\frac{1}{2}mr^2$ and a potential energy term $\frac{1}{2}K_p(\int_0^t r d\tau)^2$. Note the similarity between r, $\int_0^t r d\tau$ and $\dot{e}$, e of the previous controller, respectively. This Lyapunov function also contains the model error terms that we wish to minimize $\frac{1}{2}\tilde{\theta}^T\Gamma^{-1}\tilde{\theta}$. A new additional term is the potential energy term $\frac{1}{2}K_p e^2$, that is included in order to be able to group the error terms (e, r) in

matrix form. This in turn, is useful to provide stability guarantees, as will be seen later. As usual, we start by obtaining the derivative of the Lyapunov function (eq. 1.23b) and introducing the error dynamics equation 1.22a into it, which gives us eq. 1.24.

$$V = \frac{1}{2}mr^2 + \frac{1}{2}K_p(\int rdt)^2 + \frac{1}{2}\tilde{\theta}^T\Gamma^{-1}\tilde{\theta} + \frac{1}{2}K_pe^2 \tag{1.23a}$$

$$\dot{V} = m\dot{r}r + K_pr\int rdt + \tilde{\theta}^T\Gamma^{-1}\dot{\tilde{\theta}} + K_pe\dot{e} \tag{1.23b}$$

$$\begin{aligned}\dot{V} = &-\tilde{\theta}^T\Psi_dr - K_dr^2 - K_pr\int rdt + rm\lambda(r-\lambda e) - rB(r-\lambda e)\\ &+ r\mathsf{Err}_1 + r\mathsf{Err}_2 + K_pr\int rdt + \tilde{\theta}^T\Gamma^{-1}\dot{\tilde{\theta}} + K_pe(r-\lambda e)\end{aligned} \tag{1.24}$$

By re-organizing the last equation, we obtain:

$$\begin{aligned}\dot{V} = &\tilde{\theta}^T(\Gamma^{-1}\dot{\tilde{\theta}} - \Psi_dr) - K_dr^2 + rm\lambda(r-\lambda e) - rB(r-\lambda e)\dots\\ &+ r\mathsf{Err}_1 + r\mathsf{Err}_2 + K_per - K_p\lambda e^2\end{aligned} \tag{1.25}$$

In this equation, we can recognize the usual parameter update rule $\dot{\hat{\theta}} = -\Gamma\Psi_dr$ from the first term, where we can see that the update rule is basically the same as before but based on the desired known basis functions Ψ_d. The second term is the damping term K_dr^2. At this point, we need to make a design decision. We can use the damping term as a normal or an energy-based damping term (as described in the previous section) and work only with the remaining terms to find a control law u_r that can guarantee robust performance. The other alternative is to include the damping term as one degree of freedom more for defining the control law u_r. A last alternative is to divide the damping term into two separate terms, that could be used in both parts $K_d = K_{d1} + K_{d2}$. For the sake of continuity with previous work, we will take the last option. Therefore, we introduce the errors bounds Err_1 and Err_2 and group the terms as follows:

$$\begin{aligned}\dot{V} = &\tilde{\theta}^T(\Gamma^{-1}\dot{\tilde{\theta}} - \Psi_dr) - K_dr^2 - [rm\lambda(\lambda e - r) + rB(r-\lambda e)\dots\\ &- r\mathsf{Err}_1 - r\mathsf{Err}_2 - K_per + K_p\lambda e^2]\\ \dot{V} \leq &\tilde{\theta}^T(\Gamma^{-1}\dot{\tilde{\theta}} - \Psi_dr) - K_{d1}r^2 - r^2(K_{d2} + B - \lambda m) - er(m\lambda^2 - \lambda B - K_p)\dots\\ &- K_p\lambda e^2 + F_{co}s_1r^2 + F_{co}s_1\lambda|er| + F_{se}\left(s_1 + \frac{1}{C_s}\right)(r^2 + \lambda|er|)\end{aligned}$$

We can now group the different error terms into matrix form, in the following way:

$$\begin{aligned}
\dot{V} \leq & \tilde{\theta}^T(\Gamma^{-1}\dot{\tilde{\theta}} - \Psi_d r) - K_{d1}r^2 - \begin{bmatrix} r & e \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} r \\ e \end{bmatrix} \\
a_{11} = & K_{d2} - \lambda m + B - F_{co}s_1 - F_{se}\left(s_1 + \frac{1}{C_s}\right) \\
a_{12} = & \frac{1}{2}\left(m\lambda^2 - \lambda B - K_p - F_{co}s_1\lambda sign(er) - F_{se}\left(s_1 + \frac{1}{C_s}\right)\lambda sign(er)\right) \\
a_{21} = & a_{12} \\
a_{22} = & K_p\lambda
\end{aligned}$$

Remember that close to exponential convergence can be obtained with the nonlinear damping term K_{d1}, therefore, the task now is to choose the stiffness gain K_p, the damping K_{d2}, and the design parameter λ, so as to make the matrix with the a_{ij} coefficients positive semidefinite. For this, we should satisfy the following conditions:

$$\begin{bmatrix} a_{11} & a_{12} \\ 0 & a_{22} - \frac{a_{12}^2}{a_{11}} \end{bmatrix} \tag{1.28a}$$

$$a_{11} > 0 \tag{1.28b}$$

$$a_{22} - \frac{a_{12}^2}{a_{11}} > 0 \tag{1.28c}$$

$$K_{d2} - \lambda m + B - F_{co}s_1 - F_{se}\left(s_1 + \frac{1}{C_s}\right) > 0 \tag{1.28d}$$

$$K_p\lambda - \frac{a_{12}^2}{a_{11}} > 0 \tag{1.28e}$$

Without loss of generality, we can make $\lambda = 1$, and we could simplify our conditions in the following way:

$$\begin{aligned}
K_{d2} &> m - B + F_{co}s_1 + F_{se}\left(s_1 + \frac{1}{C_s}\right) \\
K_{d2} &\gg m - B + F_{co}s_1 + F_{se}\left(s_1 + \frac{1}{C_s}\right)
\end{aligned}$$

$$
\begin{aligned}
&a_{11} \approx K_{d2} \\
&K_p > \frac{a_{12}^2}{K_{d2}} \\
&a_{12} = \frac{1}{2}\left(m - B - K_p - F_{co}s_1 sign(er) - F_{se}\left(s_1 + \frac{1}{C_s}\right) sign(er)\right)
\end{aligned}
$$

The second condition (eq. 1.28c) gives us a constraint for the stiffness gain K_p. By assuming $K_p = Kd_2$, we would get:

$$
\begin{aligned}
&K_p > a_{12} \\
&K_p > \frac{1}{2}\left(m - B - K_p - F_{co}s_1 sign(er) - F_{se}\left(s_1 + \frac{1}{C_s}\right) sign(er)\right) \\
&K_p > \frac{1}{3}\left(m - B - F_{co}s_1 sign(er) - F_{se}\left(s_1 + \frac{1}{C_s}\right) sign(er)\right)
\end{aligned}
$$

Last equation can be simplified in the following way, to obtain an upper bound:

$$
K_p > \frac{1}{3}\left(m - B + F_{co}s_1 + F_{se}\left(s_1 + \frac{1}{C_s}\right)\right)
$$

In summary, both gains should be bigger than an upper bound for the unknown quantities:

$$
K_{d2} \text{ and } K_p > m - B + F_{co}s_1 + F_{se}\left(s_1 + \frac{1}{C_s}\right)
$$

It is clear, that these upper bounds are very conservative. An optimal way to select these gains and adaptively reduce them as the model certainty increases is part of an ongoing research work, leveraging ideas from optimal control and robust control methods like μ-synthesis. It would also be possible to learn these gains with reinforcement learning, where exploration is performed respecting the constraint bounds.

1.5 Comparison with other controllers

In this section, we present the results of two additional controllers. One of them is based on a reinforcement learning algorithm, namely, an actor-critic method (based on [8]). The other controller uses an EKF to estimate friction and inertial effects, and the control

command is the total estimate of the EKF (based on [7]). Details of the controllers can be found in the corresponding papers.

The online reinforcement learning algorithm is composed of two main blocks, an action network that produces an optimal control and a critic network that evaluates the performance of the action network. The critic network tries to approximate the Bellman equation and output the optimal cost-to-go, while the action network tracks a desired system trajectory while minimizing the cost function. Figure 1.6 shows the results of using this algorithm. It is easy to realize that such a controller could not be used in practice for learning, because learning is not safe. When it is exploring the space trying to reach its goal position, it has movements of unacceptable magnitude.

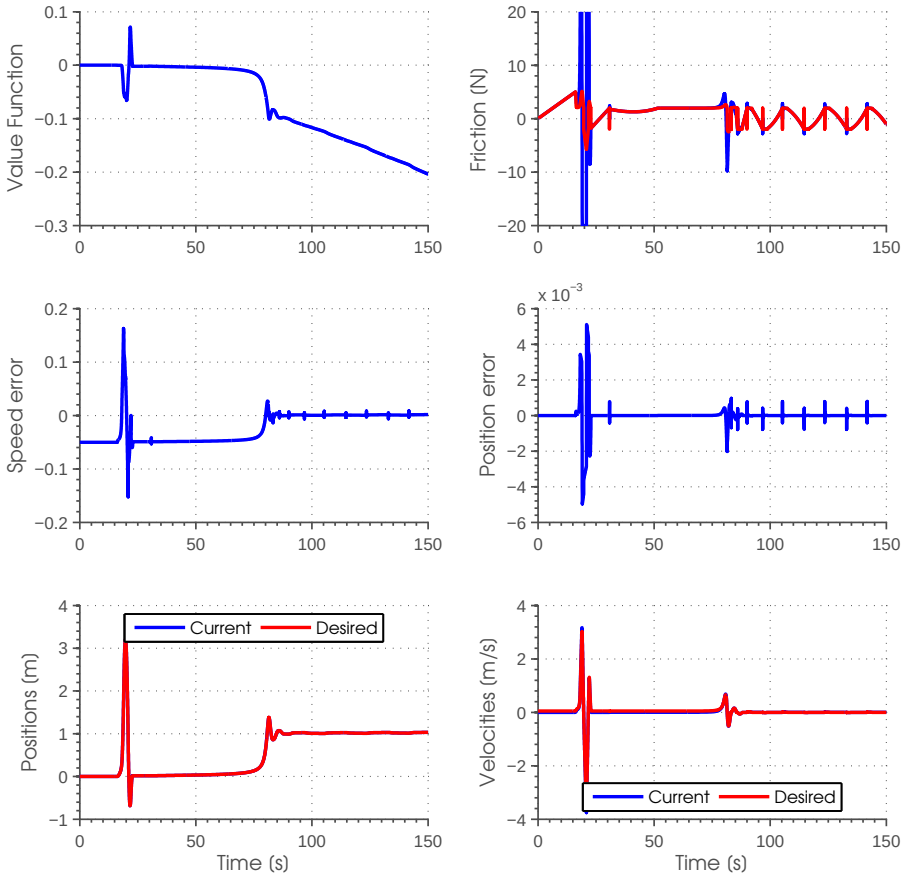


Figure 1.6: Simulation of the controller performance using the actor-critic learning algorithm.

The second controller is an Extended Kalman Filter that models the unknown inertial and friction effects as a second order random walk. It optimally estimates this effects based only on position measurements, and uses this estimate as the control law. Figure 1.7 shows the results for this controller. It seems a very nice option, because it can very quickly drive the system towards the goal and with very little overshoot. It also approximates friction very closely. One thing to keep in mind is that a Kalman Filter

is an optimal estimator, not a controller, meaning that as an estimator it will always do as good as possible with the given measurements; but this does not guarantee that the estimation can diverge, and therefore there are no convergence guarantees for the controller.

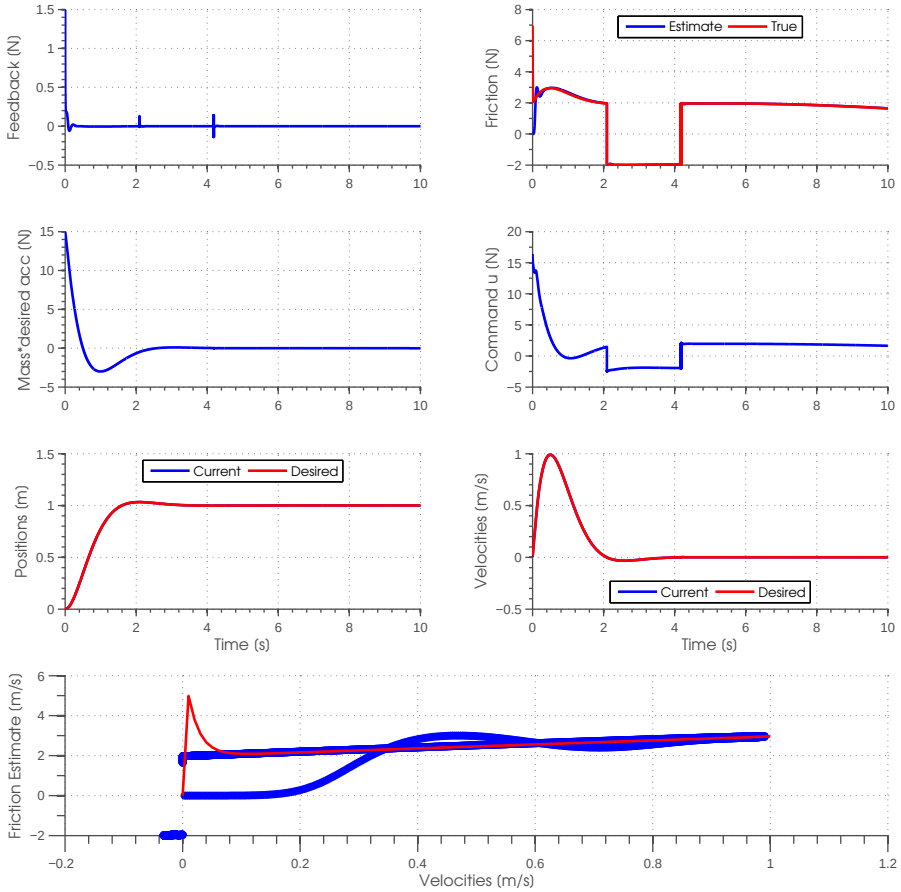


Figure 1.7: Simulation of controller based on Extended Kalman Filter.

1.6 Experiments on the KUKA robot

In this section, performance results of the adaptive controller during experiments with the KUKA robot are presented. A static model has been used for friction compensation, where the parameters are adaptively estimated. As explained in the report, a dynamic model is able to more accurately reproduce observed physical phenomena of friction, and would be a better compensator. However, in these initial experiments, only a static model is studied.

The controller has been tested in opening and closing experiments for two systems. The first system is a printer's tray that presents an abrupt transition between pre-sliding and sliding behavior. The second system is a drawer under different weight conditions.

Figure 1.8 shows results for the printer's tray opening task. To the left, results for an unloaded printer's tray are shown. To the right, the tray has been loaded with 2.5 kg of paper. The first row shows the tray's position, and it is easily seen that around second 3, the friction transition occurs and the movement suddenly starts. In the second row, the feed-forward force is plotted. As expected it increases up to some point where it exerts enough force so that the transition from elastic to plastic displacement happens and movement starts. The next rows show the adaptive controller parameters. The mass estimate for the loaded system is less than a kilogram more than that for the unloaded system, where we would have expected a change of 2.5kg.

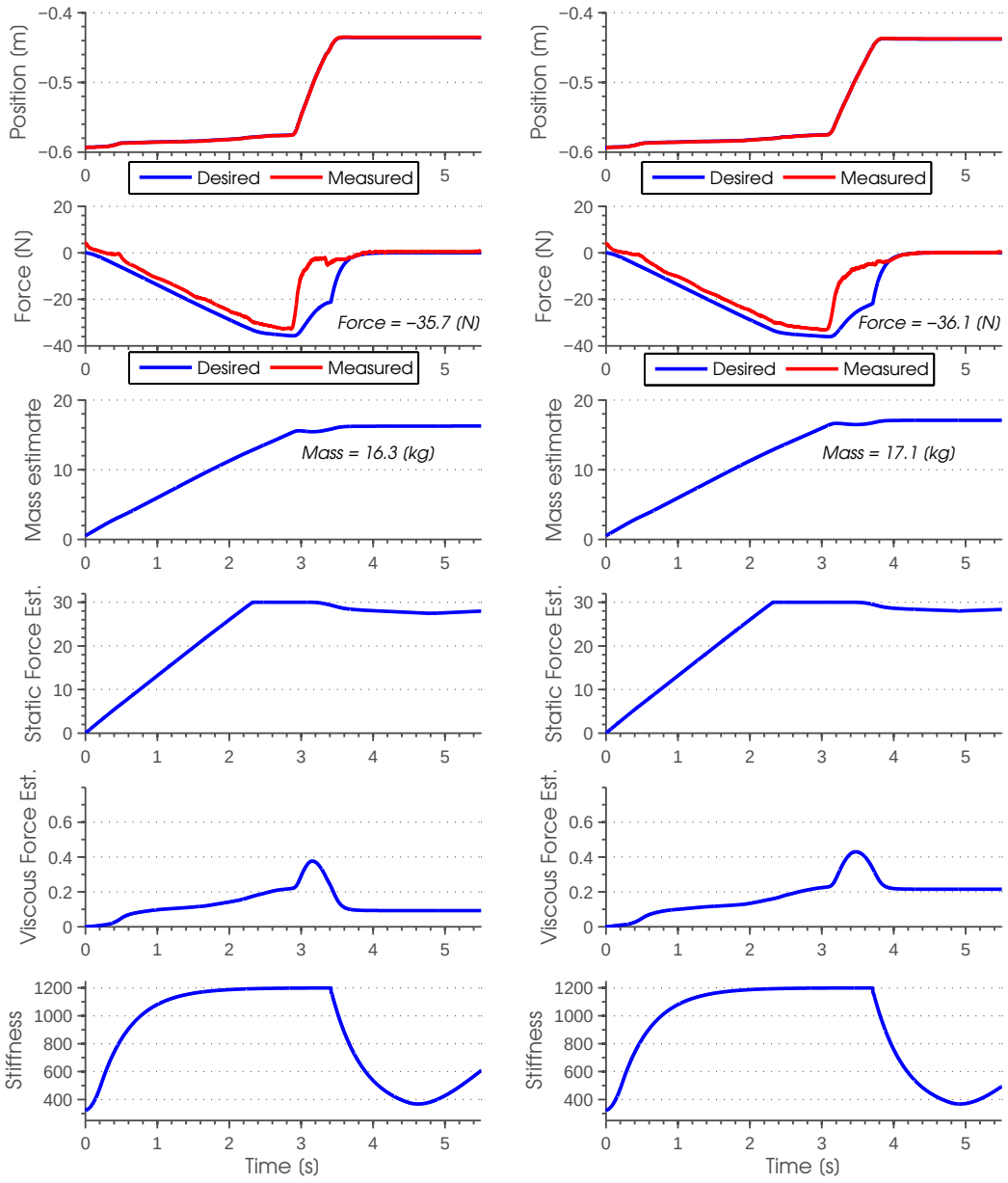


Figure 1.8: Results for the printer's tray opening experiment using the KUKA robot.

Both the coulomb friction term (Static friction in plot) and mass terms are significant for the task execution, while the force coming from the viscous friction is very small. The actual estimated viscous parameter is even smaller.

For the closing part of the experiment, it is easier to start the movement, but at the end it is necessary to exert a higher force to push the tray back into the printer. This is because of the contact between the surfaces of the tray and the printer, which increases friction opposing the motion. This is also due to the fact that a force needs to be applied so as to be sure, we are at the constraint given by the printer position, which is not exactly known for the robot. The mass estimate and Coulomb friction converge to different values than the ones seen in the opening experiment.

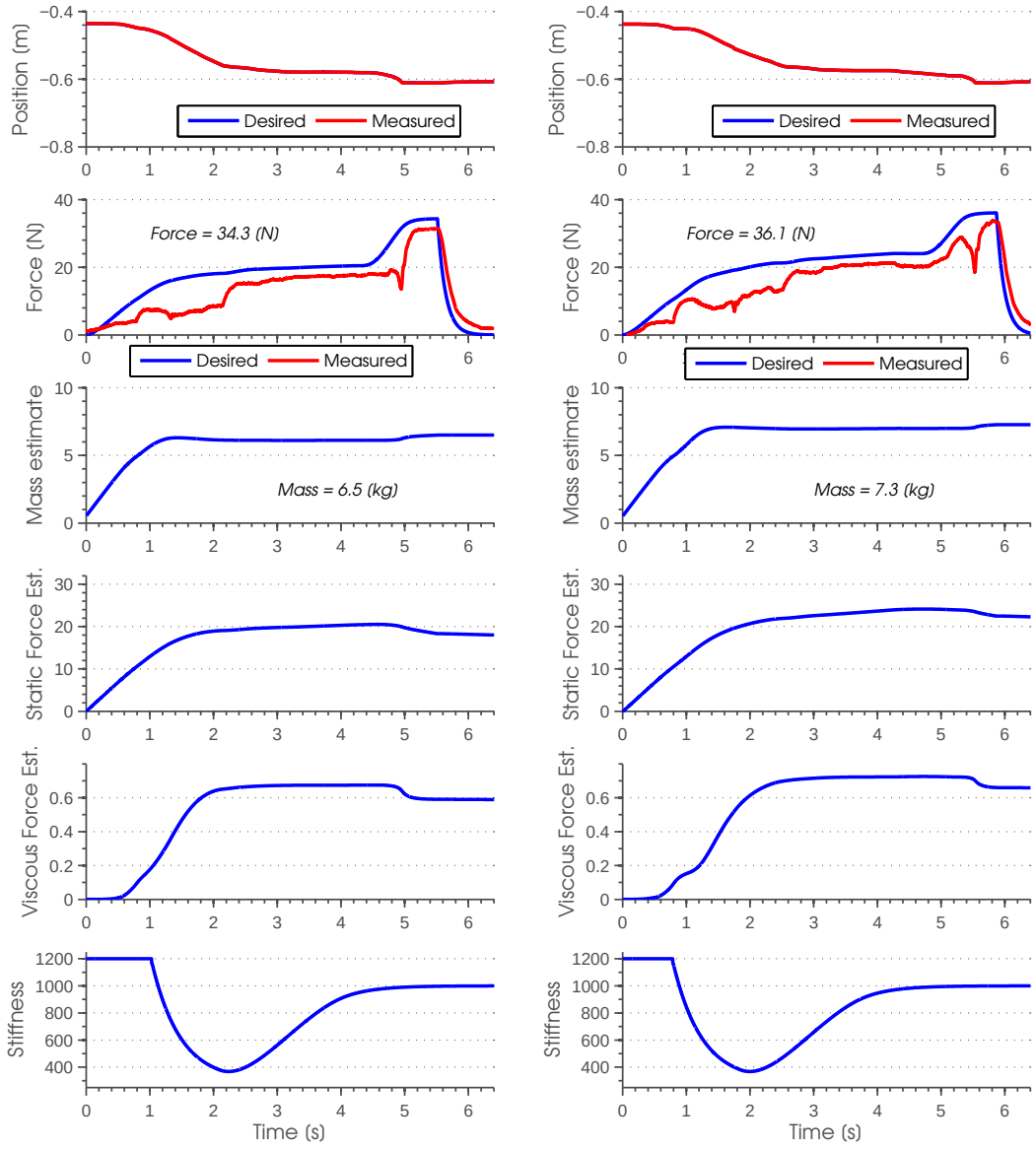


Figure 1.9: Results for the printer's tray closing experiment using the KUKA robot.

This comes from the fact that the interaction is different. In the case of opening, once motion starts, the robot rapidly reaches the final point, and the parameters have very small time to re-adapt to the new conditions. A longer interaction would allow them to reduce its values for converging to the true values. In the case of closing, the motion happens in longer time and the adaptive controller can learn the parameters under these conditions.

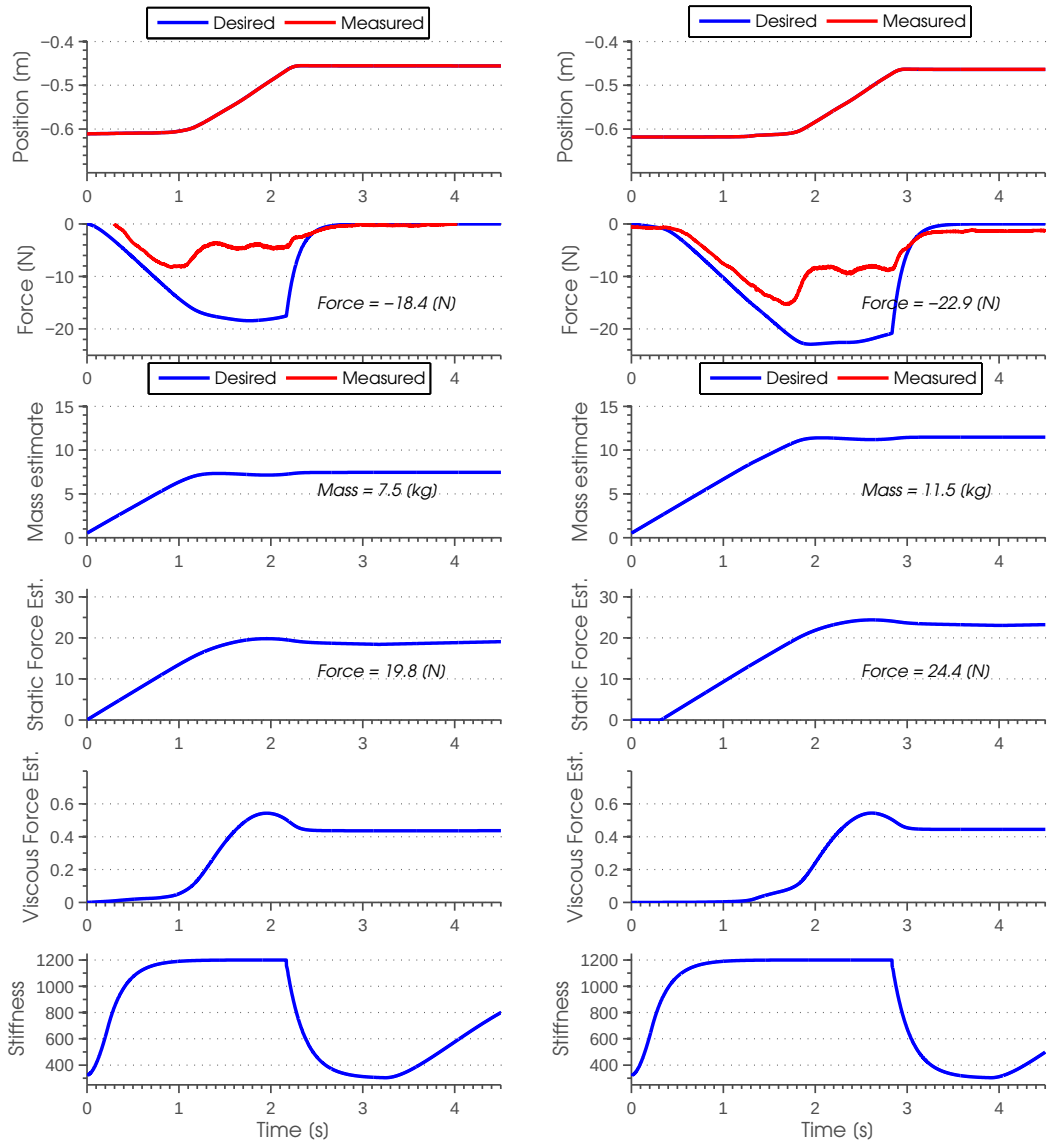


Figure 1.10: Results for a drawer opening experiment using the KUKA robot.

Finally, Figures 1.10 and 1.11 show the results for opening and closing a drawer under two different load conditions (5kg difference), respectively. The results are very similar to the ones obtained for the printer's tray. The adaptive ability of the controller makes it possible to execute the task without any parameter tuning. In this case, the mass estimation seems more consistent because opening and closing interactions are similar.

Although friction and inertial effects are not exactly estimated, the controller all together, including the compliance control is able to robustly perform the task under a large range of conditions.

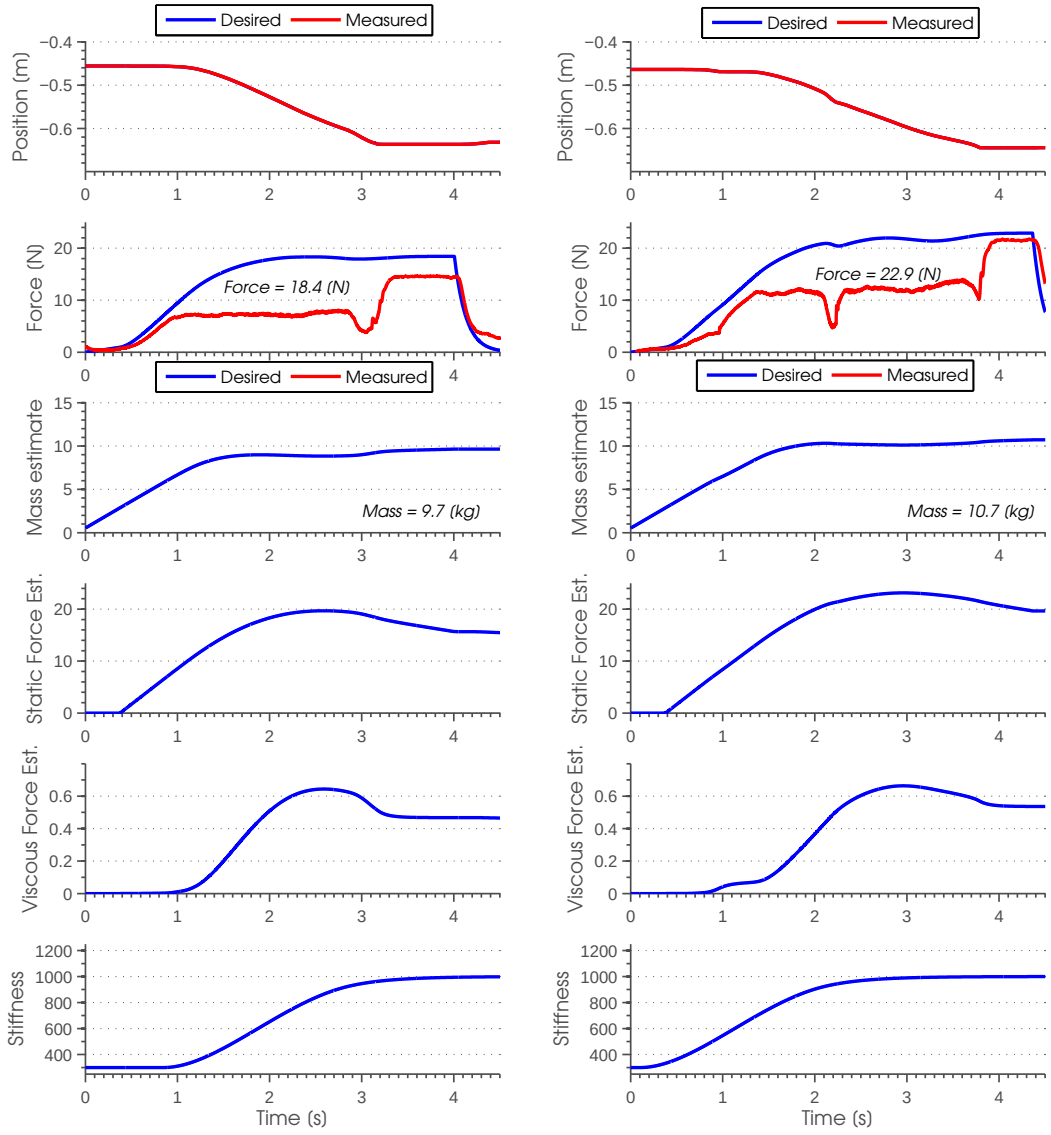


Figure 1.11: Results for a drawer closing experiment using the KUKA robot.

Bibliography

[1] Farid Al-Bender, Vincent Lampaert, and Jan Swevers. The generalized maxwell-slip model: a novel model for friction simulation and compensation. *IEEE Trans. Automat. Contr.*, 50(11):1883–1887, 2005.

[2] J. Bentsman. Oscillations-induced transitions and their application in control of dynamical systems. *ASME*, 1990.

[3] P.E Dupont. Avoiding stick-slip through pd control. *IEEE*, 1994.

[4] Pierre Dupont, Vincent Hayward, Brian Armstrong, and Friedhelm Altpeter. Single state elasto-plastic friction models, 2002.

[5] Vincent Hayward, Brian Armstrong, Friedhelm Altpeter, and Pierre E. Dupont. Discrete-time elasto-plastic friction estimation. *IEEE Trans. Contr. Sys. Techn.*, 17(3):688–696.

[6] H. Olsson, K. J. Åström, M. Gäfvert, C. Canudas De Wit, and P. Lischinsky. Friction models and friction compensation. *Eur. J. Control*, page 176, 1998.

[7] A. Ramasubramanian and Laura Ray. Adaptive friction compensation using extended kalman-bucy filter friction estimation: a comparative study. *American Control Conference*, 2000.

[8] Qinmin Yang and S. Jagannathan. Online reinforcement learning-based neural network controller design for affine nonlinear discrete-time systems. *American Control Conference*, 2007.

Chapter 5

Appendix B

Beta Process Hidden Markov Model (BP-HMM)

Sharing features using the Beta Process (Fox et al, 2009; Hughes et al, 2012): In order to handle unbounded sets of shared behaviors, the BP-HMM uses a feature-based nonparametric Bayesian approach based on the Beta Process. Each i-th time-series is represented by a sparse binary vector $\mathbf{f}^{(i)} = [f_1^{(i)}, f_2^{(i)}, ...]$ indicating feature (behavior) inclusion. Given N time series, the matrix $\mathbf{F} = [\mathbf{f}^{(i)}; ...; \mathbf{f}^{(N)}]$ indicates the binary features of the full set of time series. To induce a predictive distribution on infinite features known as the Indian Buffet Process (IBP), $\mathbf{F}$ is generated by the Beta-Process (BP). The BP depends on the realization B, which contains a set of global weights that determine the potentially infinite number of features, represented with θ_k (model parameters) and $b_k \in (0, 1)$ (inclusion in the respective i-th time-series) (Table 5.1). Each binary feature vector $\mathbf{f}^{(i)}$ is obtained using independent Bernoulli draws (BeP) parametrized by B, dependent on mass hyper-parameter γ used to determine a Poisson(γ) distribution for the number of active features in each i-th time series and concentration hyper-parameter β for controlling how often features are shared between time series.

Table 5.1: BP-HMM Generative Model

$$
\begin{gathered}
\textit{–Beta Process–} \\
B|B_o, \gamma, \beta \sim BP(\beta, \gamma B_o) \\
B = \textstyle\sum_{k=1}^{\infty} b_k \delta_{\theta_k} \\
\mathbf{f}^{(i)}|B \sim BeP(B), i = 1...N \\
\textit{–HMM Dynamics–} \\
z_t^{(i)} \sim \pi^{(i)}_{z_{t-1}^{(i)}}, \mathbf{y}_t^{(i)}|z_t^{(i)} = k \\
\eta_{jk}^{(i)} \sim Gamma(\alpha + \kappa\delta_{jk}, 1) \\
\pi_{jk}^{(i)} = \frac{\eta_{jk}^{(i)} f_k^{(i)}}{\sum_n f_n^{(i)} \eta_{jn}^{(i)}}
\end{gathered}
$$

Dynamic behavior modeling using the BP-HMM (Fox et al, 2009; Hughes et al, 2012): The derived feature model is then combined with an HMM, to create a sequential model with potentially infinite number of states. $\mathbf{f}^{(i)}$ now determines the set of states available for the i-th time-series. Each t-th time step is assigned a state $z_t^{(i)} = k$, which determines the model parameters θ_k that generated $\mathbf{y}_t^{(i)}$ (observed data of i-th time series at step t, where $\mathbf{y}_t^{(i)} \in R^d$ for a d-dimensional time series). The transition distribution $\pi^{(i)} = \{\pi_k^{(i)}\}$ is independent for each i-th time series and is derived from a normalized collection of gamma-distributed random variables $\eta^{(i)}$ with the sticky parameter κ biasing the model to match high self transitions. Hence, for each j-th state from the HMM of the i-th time-series, the transition distribution $\pi_j^{(i)}$ is built by $\eta_{jk}^{(i)}$ and $\pi_{jk}^{(i)}$, where δ_{jk} represents the active features. Furthermore, as we chose to use Gaussian emissions as the conjugate likelihood for the observed data, the model parameters for each behavior $\theta_k = \{\mu_k, \Sigma_k\}$ parametrize the observation dynamics $\mathbf{y}_t^{(i)} \sim \mathcal{N}(\mu_k, \Sigma_k)$[12].

[1]These are drawn from conjugate priors on the mean and covariance of the data, by applying the normal inverse-Wishart, defined by $(\mu, \Sigma) \sim \mathcal{NIW}(\mu_o, \lambda, \nu_o, \mathcal{S}_o)$ where ν_o are the degrees of freedom, λ is the precision of μ and $\mathcal{S}_o$ is the scaling matrix.

[2]For posterior inference, we use an improved Markov Chain Monte Carlo (MCMC) method for BP-HMM learning that rapidly discovers new and shared behaviors using split-merge moves based on sequential allocation from the Nonparametric Bayes Modeling Toolbox provided by Michael Hughes in http://michaelchughes.github.io/NPBayesHMM/

Chapter 6

Appendix C

[Full Text] Miao, Li, Yin, H., Kenji, Tahara and Billard, A. (2014) Learning Object-level Impedance Control for Robust Grasping and Dexterous Manipulation. IEEE International Conference on Robotics and Automation (ICRA), HongKong, China, May 31-June 7.

Learning Object-level Impedance Control for Robust Grasping and Dexterous Manipulation

Miao Li[1], Hang Yin[1], Kenji Tahara[2,1] and Aude Billard[1]

***Abstract*— Object-level impedance control is of great importance for object-centric tasks, such as robust grasping and dexterous manipulation. Despite the recent progresses on this topic, how to specify the desired object impedance for a given task remains an open issue. In this paper, we decompose the object's impedance into two complementary components– the impedance for stable grasping and impedance for object manipulation. Then, we present a method to learn the desired object's manipulation impedance (stiffness) using data obtained from human demonstration. The approach is validated in two tasks, for robust grasping of a wine glass and for inserting a bulb, using the 16 degrees of freedom Allegro Hand mounted with the SynTouch tactile sensors.**

I. INTRODUCTION

Robust grasping and dexterous manipulation are two of the most important capabilities that a robot is expected to have. The main characteristic of a robust grasp is its ability to comply with external perturbations applied to the grasped object while still maintaining the grasp. In dexterous manipulation, the robotic hand, mainly the fingertips, have to physically interact with the object in order to move it to a desired configuration. In both scenarios, appropriate grasping forces need to be applied on the grasped or manipulated object, either to keep the grasp stable under perturbation or to move the object to a desired configuration.

To this end, various control algorithms have been proposed and ported to control multi-fingered robotic hand. These can be roughly divided into two groups. The first group encompass hybrid position/force control approaches that modulates the force explicitly to manage the interaction imposed by the environment [1], [2], [3]. Another group uses impedance control to regulate the interaction force implicitly by specifying the impedance of the grasped object [4], [5], [6]. [7]. In general, the hybrid position/force control is more precise when controlling simultaneously the force and position. The main deficiency of hybrid control is the transition between position and force control when the contact state varies between non-contact and contact. A small delay in this transition may lead to a very large overshot contact force. In addition, the selection of accurate grasping forces for hybrid control that fulfil the friction constraints and task requirements is still a difficult planning problem [3]. In impedance controller, the object motion is realized by a desired object impedance that generates force to move the object to a desired configuration. It has the advantage that it will converge to the desired position in free motion and a stable equilibrium position in the case of interaction with the environment. This merit can be greatly beneficial for both robust grasping and dexterous manipulation. Therefore, we restrict the rest of this review to impedance controllers only. In [8], a fingertip Cartesian stiffness controller was introduced using fingertip force sensor. However, the stiffness controller can not actively control the whole system dynamics. To overcome this defect, Liu and Hirzinger [9] proposed a Cartesian impedance controller for the DLR hand based on the joint torque measurements. While these two controllers are in the fingertip Cartesian space, object-level impedance controllers are proposed by directly specifying the desired impedance in the object frame, which are usually more suitable for robust grasping and dexterous manipulation of an object. In [4], an object level impedance controller has been proposed for a multi-arm manipulator to directly control the internal object forces and compensate the system dynamics. The object is assumed to be rigidly grasped that can transmit bilateral contact forces between the fingertips and the object. Wimbock et al. [5], [7] recently presented their experimental evaluation of an intrinsically passive controller for multi-fingered hand, where the damping parameters are designed and implemented as a function of the object effective inertia and stiffness matrix. A similar impedance controller was also proposed in [6] and [10] by defining a virtual frame which depends only on the fingertip positions. The damping parameters are designed in

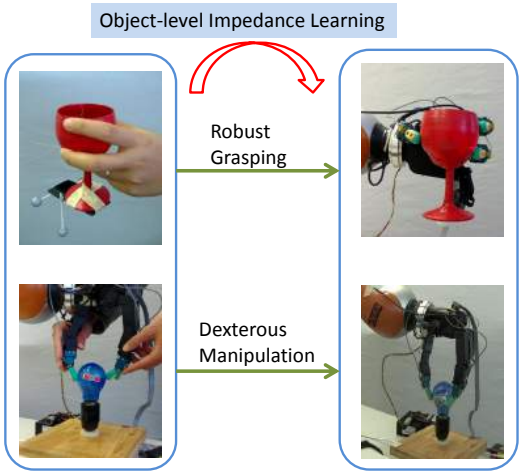


Fig. 1: The obejct-level impedance for robust grasping and dexterous manipulation are learned from human demonstration.

[1]M. Li, H. Yin and A. Billard are with LASA, École Polytechnique Fédérale de Lausanne (EPFL), Switzeland {miao.li, hang.yin,aude.billard}@epfl.ch

[2]K. Tahara is with Faculty of Engineering, Kyushu University, 744 Moto'oka, Nishi-ku, Fukuoka 819-0395, Japan. He is currently a visiting scholar at LASA. tahara@ieee.org, kenji.tahara@epfl.ch

Preprint submitted to 2014 IEEE International Conference on Robotics and Automation. Received September 15, 2013.

both the finger joint space and the object frame.

However, despite all the above-mentioned progress, one critical issue still remains unaddressed: how to specify the proper impedance for a given task? The specification of impedance is known as a difficult problem as it depends on the task at hand as well as the kinematic and dynamic limitation of the robot [11], [12]. Moreover, the impedance parameters may need to adapt to the task requirements or to variation in the environment, such as the bulb replacement task that the torsional resistance increases greatly during the last phase of the task. To this end, sensor feedback should be taken into account to monitor the status of task's completion and to vary the impedance accordingly.

In this paper, we attempt to address this problem by learning the impedance from human demonstration. In the following Section II, some related works regarding impedance specification are summarized. In Section III, an object-level impedance controller is reformulated. In Section IV, methods for learning impedance from human demonstration for robust grasping and dexterous manipulation are presented. Experiments on a multi-fingered robotic hand are demonstrated and discussed in Section V. Finally, we give a conclusion and an outlook on future work in Section IV.

II. RELATED WORK

a) Analytical Impedance Specification: In one of their early works, Mason and Salisbury [8] used the congruence transformation to obtain the desired object stiffness from joint stiffness. Based on this work, Cutkosky and Kao [13] expressed the compliance of a grasp as a function of grasp geometry, contact conditions and mechanical properties of the fingers. In order to choose the grasp compliance for a given task, Shimoga and Goldenberg [14] formulated a concept termed Grasp Admittance Center, which is the origin of a frame that impedance matrices will be diagonal. Also, A qualitative method has been developed to choose the relative magnitude of the impedance parameters for a set of tasks. In [15], [16], Kim et al. analysed the compliance characteristics for different tasks by considering the grasp geometry, which is the relation between the operational space and the fingertip space of multi-fingered hand. Their analytical results show that the non-diagonal terms in impedance matrices can not be specified arbitrarily and they also used a qualitative method (small and large value of stiffness) to specify the impedance parameter.

b) Impedance Learning: Learning of tasks is another approach by which desired impedance parameters can be specified. In [17], the impedance learning problem is formulated as a model-based reinforcement learning problem, where the impedance parameters can gradually change to improve the task performance. In [18], the authors accomplished a variable impedance controller with a model-free, sample-based reinforcement learning method. However, the reinforcement function needs to be carefully defined to capture the essence of the task, which will be difficult for complex tasks, such as dexterous manipulation. Sikka and McCarragher [19] presented a method that can learn the robot end-point stiffness of contact tasks from human demonstration. An online, incremental algorithm has been proposed in [20] to learn varying end-point stiffness from human demonstration. For a multi-fingered robotic hand, a implicit compliant controller [21] is learned to adapt the grasp under perturbation, which actually mapping the fingertips tactile response to finger joints. However, this method is hand dependent and difficult to generalize to manipulation tasks.

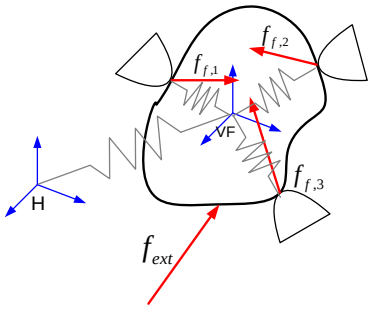


Fig. 2: An object grasped by 3 fingers. The object impedance and grasp impedance are shown as springs. The $\mathbf{f}_{f,i}, i = 1, 2, 3$ are the contact forces on each fingertips. $\mathbf{f}_{ext}$ is the external perturbation force. The frame **H** and **VF** are the inertial frame and the virtual frame, respectively.

As discussed in [22], the tasks for multi-fingered hand are usually object-centric. In these cases, learning an object-level impedance is more suitable and can be easily applied to other hands. In this paper, we extend the object-level impedance controller in [10] with tactile feedback and object-level impedance learned from human demonstration.

III. OBJECT-LEVEL IMPEDANCE CONTROL

In this section, we will reformulate the object-level impedance controller proposed in [10], which is mainly composed of two parts, a stable grasp controller and an object manipulation controller.

A. Object Manipulation Impedance

Following the formulation of impedance control in [11], the dynamics of the object, as shown in Fig. 2, is governed by the equation:

$$\mathbf{f}_{f,o} + \mathbf{f}_{ext} = M_0 \ddot{\mathbf{x}} \tag{1}$$

where $\mathbf{f}_{f,o}$ is the summation of manipulating forces $\mathbf{f}_{f,oi}$ exerted on the object from each fingertip, $\mathbf{f}_{ext}$ is the external perturbation force. All the forces are expressed in the inertial frame. M_0 is the actual inertia matrix and $\mathbf{x}$ is the position and orientation of the object. Usually, the position and orientation are controlled independently [10]. Here for simplicity, we put position and orientation in one vector $\mathbf{x}$ to introduce the controller.

The objective of impedance control is to modulate the interaction between the object and the environment by controlling the contact forces. The desired interaction of the system is given by:

$$\mathbf{f}_{ext} = M\ddot{\mathbf{x}} + D(\dot{\mathbf{x}} - \dot{\mathbf{x}}_r) + K(\mathbf{x} - \mathbf{x}_r) \tag{2}$$

where $\mathbf{x}_r, \dot{\mathbf{x}}_r$ is the reference trajectory and M, D, K are the desired apparent inertia, damping and stiffness, respectively. From equation (1) and (2), we can derive the object-level impedance control law given as:

$$\mathbf{f}_{f,o} = ED(\dot{\mathbf{x}}_r - \dot{\mathbf{x}}) + EK(\mathbf{x}_r - \mathbf{x}) + (E - I)\mathbf{f}_{ext} \quad (3)$$

where $E = M_0 M^{-1}$ and I is the identity matrix. In practice, it is often sufficient to keep the inertia unchanged, i.e., $M_0 = M$ and only shape the stiffness and damping. Then the equation (3) can be simplified as:

$$\mathbf{f}_{f,o} = D(\dot{\mathbf{x}}_r - \dot{\mathbf{x}}) + K(\mathbf{x}_r - \mathbf{x}) \quad (4)$$

B. Stable Grasping Impedance

Up to now, only the object manipulation impedance has been considered. In order to make the grasp stable during manipulation, we need to design a stable grasping impedance, which can be used to change the grasping forces. In our paper, the contact model between the object and the fingertips is assumed to be point contact with friction, which can only transmit contact forces. Therefore, we only use one translational spring connecting each fingertip and the origin of the object (virtual) frame to represent the stable grasping impedance (stiffness), as shown in Fig. 2. The grasping forces can be expressed as:

$$\mathbf{f}_{f,gi} = K_{gi}(\|\Delta\mathbf{p}_i\| - L_i)\frac{\Delta\mathbf{p}_i}{\|\Delta\mathbf{p}_i\|} \quad (5)$$

where $\mathbf{f}_{f,gi}$ and K_{gi} are the grasping force and stable grasping stiffness at i-th fingertip. $\Delta\mathbf{p}_i = \mathbf{p}_o - \mathbf{p}_i$ with $\mathbf{p}_i$ as the position of contact point on i-th fingertip and $\mathbf{p}_o$ as the position of the object frame origin. L_i is the desired distance from the i-th fingertip to the object frame origin.

C. Implementation Issues

A rigorous implementation of the controller will require a lot of computational load [23]. To reduce it, the finger dynamics is not compensated and thus joint torques at each finger can be obtained from a simple Jacobian transpose.

$$\boldsymbol{\tau}_{f,i} = J_{f,i}^T \mathbf{f}_{f,i} \quad (6)$$

where $\boldsymbol{\tau}_{f,i}$ are the joint torques at i-th finger and $J_{f,i}$ is the Jacobian of the i-th finger. The contact force can be computed as: $\mathbf{f}_{f,i} = \mathbf{f}_{o,i} + \mathbf{f}_{g,i}$. The more rigorous way to compute the contact force using grasp mapping can be also use here [7], which is also more computational expensive.

In order to implement this controller, we need to address the following issues: (1) measure the object position and orientation $\mathbf{x}$; (2) design the reference trajectory $\mathbf{x}_r, \dot{\mathbf{x}}_r$; (3) choose the impedance parameters K and D. While (1) will be discussed in the remaining part of this section by introducing a *Virtual Frame*, the method to deal with (2) and (3) by learning from human demonstration will be presented in the next section.

D. Virtual Object Frame

Due to the occlusion of the hand, it is still very difficult to rely on vision to obtain the actual object position and orientation in the controller. To deal with this, the concept of Virtual Frame (VF) is adopted here, which is a function of all the contact points between object and fingertips. Virtual frame (VF) can be used to estimate the real object position and orientation if we assume that relative contact points between object and fingertips do not change [1]. Different from the definition in [10], the VF in this work is the function of real contact point on each fingertip, which can be obtained from tactile feedback. In our work, we only use three fingers, the origin of VF is:

$$\mathbf{p}_o = \frac{1}{3}\sum_{i=1}^{3}\mathbf{p}_i \quad (7)$$

The orientation of the frame is defined in the following way:

$$\begin{aligned} R_o &= [\mathbf{r}_x, \mathbf{r}_y, \mathbf{r}_z] \in SO(3) \quad (8)\\ \mathbf{r}_x &= \frac{\mathbf{p}_3 - \mathbf{p}_1}{\|\mathbf{p}_3 - \mathbf{p}_1\|}\\ \mathbf{r}_z &= \frac{(\mathbf{p}_3 - \mathbf{p}_1) \times \mathbf{r}_x}{\|(\mathbf{p}_3 - \mathbf{p}_1) \times \mathbf{r}_x\|}\\ \mathbf{r}_y &= \mathbf{r}_z \times \mathbf{r}_x \end{aligned}$$

With the defined VF, one can compute the translation and rotation difference between the VF and the desired or reference frame. Thus, from equation (4), (5) and (6), the desired joint torque for each finger can be calculated. For more details about the implementation, one can refer to [10].

IV. IMPEDANCE LEARNING FROM HUMAN DEMONSTRATION

In this section, methods about how to specify impedance for robust grasping and dexterous manipulation will be presented.

A. Relative Impedance for Robust Grasping

A robust grasp should be able to comply with external perturbation from any directions. But in different directions, the extent of compliance will depend on the grasp configuration as well as the task requirement. For instance, grasping a screwdriver as a tool and grasping a pen to write will require totally different levels of rotational compliance along the axial direction.

Our method of impedance selection for robust grasping is quite intuitive: the object stiffness in one direction is inversely proportional to the variance of displacement under perturbation in the corresponding direction. From this assumption, we can learn the relative stiffness for robust grasping in different directions from human demonstration. This idea has also been utilized to learn the end-point stiffness for a single manipulator [20]. During the demonstration, an object is grasped by a human demonstrator with eyes closed. To mimic the fact that our controller will use solely

[1]This assumption will neglect the rolling and slipping effects.

proprioceptive and tactile information, with no vision. The grasped object is perturbed by another person randomly and the displacement of the object is recorded $\{\mathbf{x}^i, i = 1...N\}$. Then the object stiffness can be specified as follows:

$$K = \alpha\{\frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}^i - \mathbf{x}_r)(\mathbf{x}^i - \mathbf{x}_r)^T\}^{-1} \quad (9)$$

where $\alpha \in \mathbb{R}^+$ is a ratio parameter that needs to be set manually and $\mathbf{x}_r \in \mathbb{R}^6$ is the object initial (and desired) position and orientation.

Besides stiffness specification, object workspace modelling is also very important for robust grasping as it determines the extent of motion of a grasped object during perturbation. However, the workspace of a grasped object will depend on the hand kinematics and the grasp configuration. To this end, we teach the robot the extend to which it can stretch its fingers through kinaesthetic demonstration, by back-driving the fingers, see Fig. 3. All positions and orientations adopted by the hand during the demonstration are used to build a probabilistic model of the workspace of the hand. The use of a probabilistic model is advantageous as it accounts for the imprecision of the recording and allows to generalize outside the demonstrations. The latter is particularly important since demonstrations may not be exhaustive and may not explore all possible postures. Here, we use Gaussian Mixture Model (GMM). A GMM is a probabilistic model of density function composed of K Gaussian components. The likelihood of each position/orientation under this model is given by:

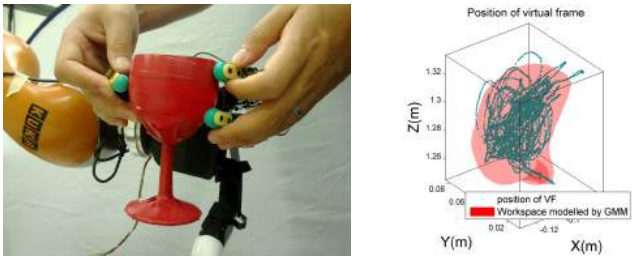
$$p(\mathbf{x}) = \sum_{k=1}^{K}\pi_k\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \quad (10)$$

where π_k is the prior of the kth Gaussian component and $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is the Gaussian distribution with mean $\boldsymbol{\mu}_k$ and covariance $\boldsymbol{\Sigma}_k$. A new VF, computed using Eq. (7) (8), is said to lie in the object's workspace if its likelihood to belong to the model is greater than a fixed threshold, i.e. $p(x_*) > L_{thresh}$. This threshold, in our experiment, is quite conservative and is set as no more than 2 standard deviation, which means that about 95.45% training position/orientation of VF will be covered by the learned GMM. For more details about the parameters selection for training GMM, one can refer to [24].

With the object workspace model, one can design a desired reaction behavior to improve grasping stability, eg., increase object stiffness gradually when the object is approaching boundary of learned working space. This could be achieved in our model by increasing the ratio parameter α in equation (9). This would however increase stiffness by the same amount in all directions. It may often be useful to be able to shape this increase along particular directions, such as the direction the moves the object farthest away from the workspace's boundary.

B. Variable Impedance for Dexterous Manipulation

In the case of robust grasping, the reference frame can be easily set as the initial position and orientation, which does not vary with time. For dexterous manipulation, a time-varying reference trajectory $\mathbf{x}_r, \dot{\mathbf{x}}_r$ will be required. In this section, we will present a method that learns the reference trajectory and the desired object impedance simultaneously.



(a) (b)

Fig. 3: (**a**) Human teaching of object workspace. The object impedance are set to zero in all directions during the demonstration, which means human can move the object freely in its workspace. (**b**) The position and orientation of VF are recorded and trained using GMM. Here is shown the trained result in the subspace of VF position. The red surface is the iso-surface with the same threshold likelihood L_{thresh}

The objective of human demonstration is to model the interaction between the object being manipulated and the environment. Thus, during the demonstration, at each sample instant $i, i = 1...N_s$, the motion of the object $\{\mathbf{x}(i), \dot{\mathbf{x}}(i)\}$ and the sum of manipulating forces $\mathbf{f}_{f,o}(i)$ applied on the object are recorded[2]. Consider $t = 1...N_t$ consecutive samples of data obtained over a short time window. Assuming the impedance parameters and reference trajectory remain constant over this time window, the relationship between the object motion and the force exerted on object is given by:

$$\mathbf{f}_{f,o}(i) = D(\dot{\mathbf{x}}_r - \dot{\mathbf{x}}(i)) + K(\mathbf{x}_r - \mathbf{x}(i)), i = 1...N_t; \quad (11)$$

During each time window, since we assume that the object's impedance parameters and the reference trajectory are not changing with time, they can be obtained by minimizing the following objective function:

$$\min_{D,K,\dot{\mathbf{x}}_r,\mathbf{x}_r}: \sum_{i=1}^{N_t}\|\mathbf{f}_{f,o}(i) - \{D(\dot{\mathbf{x}}_r - \dot{\mathbf{x}}(i)) + K(\mathbf{x}_r - \mathbf{x}(i))\}\|^2 \quad (12)$$

In practice, the term from damping is usually ignored by assuming that the desired velocity trajectory is the same as the measured one and thus equation (12) can be simplified as:

$$\min_{K,\mathbf{x}_r}: \sum_{i=1}^{N_t}\|\mathbf{f}_{f,o}(i) - \{K(\mathbf{x}_r - \mathbf{x}(i))\}\|^2 \quad (13)$$

One should note that the assumption that object's impedance parameters and reference trajectory are stationary for short period of time only works for slow task right now. For fast task, it may require to use a high speed (force and motion) sensor to collected enough to obtain a reasonable optimal solution for eq. (13). Also, from eq. (13), we can obtain

[2]In practice, only the contact forces on each fingertip can be measured, which include the grasping forces and the manipulating forces. the sum of grasping forces is very small in our setting, i.e., equation (5), which can be ignored.

the desired impedance parameters for each time window and their corresponding reference trajectory. In this framework, the desired impedance parameters and the reference trajectory will depend on time. To account for this, we define a variable $\phi \in [0, 1]$ to represent the completion of the task, which is a function of the desired trajectory, i.e., $\phi = \Phi(\mathbf{x}_r)$. In our experiments, Φ is given by the distance from the current configuration to the goal configuration. Thus, the impedance parameters and desired trajectory are expressed as a function of ϕ.

Since the system should be stable, additional constraints should be taken into account. First, the stiffness matrix should be positive semi-definite and its elements must be less than some maximum value since we assume that human will not demonstrate extremely large object stiffness. Also, for the reference trajectory, it should be close to the actual measured object trajectory. Thus we have:

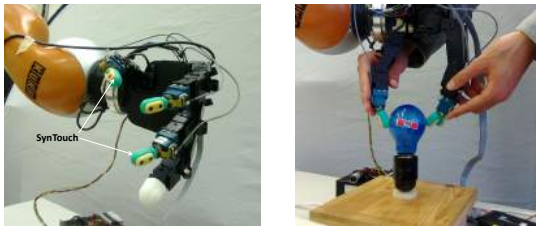
$$\begin{aligned} &K_{i,j} \leq k_{lim}, \quad i = 1...6, j = 1...6; \\ &\|\mathbf{x}_r - \mathbf{x}(i))\| \leq \Delta x_{lim}, \quad i = 1...N_t; \\ &\|\dot{\mathbf{x}}_r - \dot{\mathbf{x}}(i)\| \leq \Delta \dot{x}_{lim}, \quad i = 1...N_t; \end{aligned} \tag{14}$$

where $\Delta x_{lim} \in \mathbb{R}^+, \Delta \dot{x}_{lim} \in \mathbb{R}^+$ is upper bound of the difference between the actual and real (position and velocity) trajectories. With the objective function (13) and the constraints (14), the optimized impedance parameters and the reference trajectory can be obtained in each time window.

V. EXPERIMENTS AND DISCUSSION

In the experiments, we use a 4-fingered Allegro hand[3] to test the object impedance specification for robust grasping and dexterous manipulation. The initial grasp and the grasping stiffness are predefined.

A. Setup



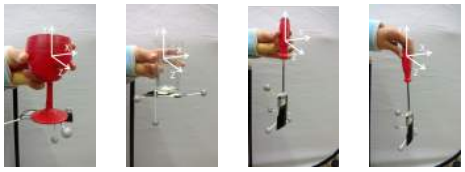
(a) Allegro hand (b) Human demonstration

Fig. 4: (**a**) The Allegro hand mounted with the SynTouch tactile sensors on the fingertips; (**b**) Human demonstration of bulb replacement.

Each of the four fingers of the *Allegro hand* has 4 independent torque-controlled joints, see Fig. 4a. In our experiments, we only use 3 fingers even though our controller can be generalized to 4 fingers. Each fingertip of these 3 fingers has been mounted with a biometric tactile sensor from *SynTouch*[4], which has been calibrated to provide contact information such as contact position and contact force.

[3]http://www.simlab.co.kr/Allegro-Hand.htm

[4]http://www.syntouchllc.com/



(a) (b) (c) (d)

Fig. 5: Human demonstration of robust grasping on 3 different objects: glass, cup, screwdriver (side and top grasp). The motion of the object when perturbed is tracked by OptiTrack.

B. Robust Grasping

In the robust grasping experiment, a human expert demonstrates 4 grasps as shown in Fig. 5. The arm and wrist are fixated on the table so that the object motion will only come from the finger motion. During the experiments, for each object the perturbations are applied by another person randomly. The position and orientation for the objects are tracked using a motion capture system from *OptiTrack*[5] at a sampling rate of 240Hz. More than 10000 datapoints are collected for each object.

The recorded object orientation is transformed into RPY Euler angles. The relative impedance parameters for the 4 grasps in different directions are computed using equation (9). In general, the choice of frame of reference depends on the task. Here we compute a diagonal stiffness matrix in the reference frame of the object since this is also the frame of reference in our impedance controller. It is also possible to extract the principle directions and corresponding stiffness along these directions from eq. (9) [20], but then we need to transform the stiffness along these principle directions into the object's frame of reference in real time during implementation.

The relative stiffness for these grasps are shown in Fig. 6–Fig. 9. The results show that in different directions, the relative stiffness is indeed different, which means that an isotropic stiffness is not always suitable. Also, comparing the relative stiffness for two different grasps on the screwdriver (Fig. 5, (c) and (d)), see Fig. 8 and Fig. 9, we found that the rotational stiffness around Y-axis is totally different. In the top grasp, the rotational stiffness around Y-axis is much smaller than that of side grasp, which means that the top grasp requires smaller forces to rotate the screwdriver around Y-axis. This result coincides with our intuition. Here, we only show the implementation results from the grasp on the glass. The parameters are set as follows: $K_{gi} = 20N/m$, $L_i = 0.5\|\Delta\mathbf{p}_i\|m$, $k_{tx} = 20N/m$, $k_{ty} = 240N/m$, $k_{tz} = 30N/m$, $k_{rx} = 1.2 \times 10^{-3}Nm/\deg$, $k_{ry} = 6 \times 10^{-3}Nm/\deg$, $k_{tz} = 1.2 \times 10^{-3}Nm/\deg$. The snapshot of the implementation on the Allegro hand is shown in Fig. 10.

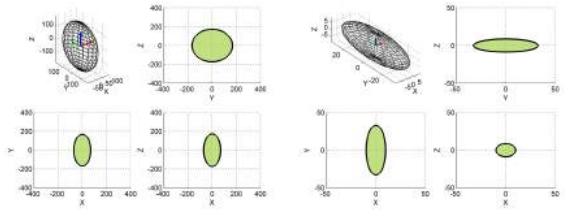
C. Dexterous Manipulation

For dexterous manipulation, we use the bulb replacement as an example, Fig. 4b. The bulb is initially on the socket

[5]http://www.naturalpoint.com/optitrack/

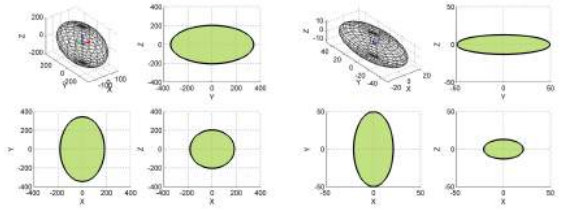
(a) (b) (c) (d) (e)

Fig. 10: Testing of robust grasping: Snapshots of the response of our controller when a human perturbs the original position of the glass. The fingers adapt smoothly to follow the direction of motion induced by the human. The impedance was learned from former human demonstration, using results in Fig. 6. The video is available at: `http://lasa.epfl.ch/~miao/robust_grasping.wmv`



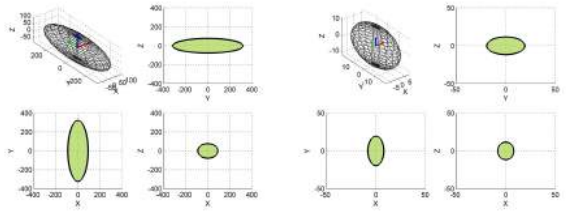
(a) glass (b) glass

Fig. 6: (**a**): The relative translational stiffness for glass, $k_{tx} < k_{tz} < k_{ty}$; (**b**): The relative rotational stiffness for glass, $k_{rx} \approx k_{rz} < k_{ty}$.



(a) cup (b) cup

Fig. 7: (**a**): The relative translational stiffness for cup, $k_{tx} \approx k_{tz} < k_{ty}$; (**b**): The relative rotational stiffness for cup, $k_{rx} < k_{rz} < k_{ty}$.

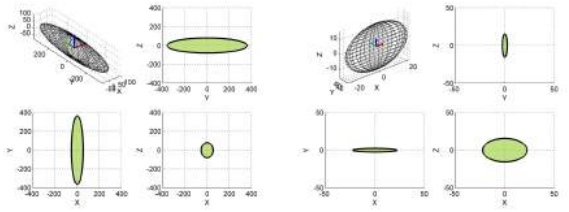


(a) screwdriver(side) (b) screwdriver(side)

Fig. 8: (**a**): The relative translational stiffness for screwdriver (side grasp), $k_{tx} \approx k_{tz} < k_{ty}$; (**b**): The relative rotational stiffness for screwdriver (side grasp), $k_{rx} \approx k_{rz} < k_{ty}$.

already. During the human demonstration, only two fingers are used as the impedance is learned in object's frame of reference and using two fingers is easier to demonstrate. The manipulating forces are measured using SynTouch mounted on the fingertips. The object real trajectory is tracked using OptiTrack [6]. Using equation (13) and (14), with $\Delta x_{lim} =$

[6]During the human demonstration, in order to track the object robustly, the experimenter must take care of not placing his fingertips on the vision markers.



(a) screwdriver(top) (b) screwdriver(top)

Fig. 9: The relative translational stiffness for screwdriver (top grasp), $k_{tx} \approx k_{tz} < k_{ty}$; (**b**): The relative rotational stiffness for screwdriver (top grasp), $k_{ty} < k_{rx} \approx k_{rz}$.

$60\,\text{deg}$, $k_{lim} = 100N.mm/deg$[7], the reference trajectory and desired stiffness for bulb replacement are obtained and shown in Fig. 11 and Fig. 12a, respectively.

If we compare the desired rotation angle with the actual rotation angle, we see that the difference varies during the whole task. This means that human demonstrator indeed regulates the difference between the actual and reference trajectories as well as the stiffness parameter. When looking at the desired object stiffness, Fig. 12a, we see that the desired stiffness increases significantly during the last phase of the task. This is due to the fact that the resistance torque between the bulb and the socket increases significantly at the last phase. We repeated this demonstration 10 times, the obtained desired stiffness for each trial is shown in Fig. 12b.

If we could measure the rotational angle of the bulb using vision, then the status of task completion ϕ and the corresponding k and $\mathbf{x}_r$ can be obtained directly. Unfortunately, it is difficult to rely on vision for dexterous manipulation task as the hand often obstructs the object from the camera's view. For this reason, we rely on tactile information to guide the task process. Figure 12 indicates that the regulation of stiffness during this task seems to follow two distinct phases. During the first phase, which occurs before break point $\phi = 0.8$ (i.e. more than 2/3rd of the total duration of the task), the stiffness is quasi constant. Whereas in the second phase, it increases steadily. We model this by setting a constant value for the stiffness for the first phase and by increasing linearly the stiffness up to its upper bound for the remainder of the task, i.e., $4N.mm/deg$, see Fig. 12b. We noticed during the implementation that this breakpoint corresponds to the instant when one fingertip (usually the thumb of Allegro

[7]Δx_{lim} is chosen by considering the rotation limitation of human hand and the Allegro hand.

hand) starts slipping on the bulb. In order to detect the slippage, we use the contact forces $\mathbf{f}_c = [f_{cx}, f_{cy}, f_{cz}]$ from SynTouch on each fingertip, with f_{cx}, f_{cy} and f_{cz} being the tangential forces in two directions and the normal force, respectively. A slippage occurs at one fingertip if the contact forces that fingertip satisfying $\sqrt{f_{cx}^2 + f_{cy}^2} > \mu f_{cz}$, μ is the coefficient of friction that is set manually. In our task, we choose $\mu = 0.9$. Fig. 14 shows the computed coefficient of friction during one successful implementation. The resulting control strategy is given in Algorithm 1. The snapshot of the implementation on Allegro hand is shown in Fig. 13

Algorithm 1: Controller for bulb replacement task

```
Move fingers to initial positions: InitialGrasp();
repeat
    Impedance Control Mode:
    SetGrasp();
    Compute the VF (eq.(7));
    Set parameters:
    L_i = 0.5||Δp_i||m, k_tx = k_ty = k_tz = 0N/m
    k_rx = k_rz = 0Nm/deg, k_ry = 1 × 10^-3 Nm/deg,
    K_gi = 12N/m
    x_r = 60 deg
    interpolate x_r to smooth the controller:
    for i=1 to 1000 do
        Compute the current reference point:
        x_cr=Slerp(x_r,i);
        Send joint torques: ObjImp();
    Open Finger and move back to initial grasp:
    InitialGrasp();
until DetectSlip()
if DetectSlip() then
    Rotate the bulb for another 4 times:
    for i=1 to 4 do
        Impedance Control Mode:
        SetGrasp();
        Compute the VF (eq.(7));
        Set parameters:
        L_i = 0.5||Δp_i||m, k_tx = k_ty = k_tz = 0N/m
        k_rx = k_rz = 0Nm/deg,
        k_ry = 1 + i * 0.75 × 10^-3 Nm/deg,
        K_gi = 12 + i * 2.5N/m
        x_r = 60 deg
        interpolate x_r to smooth the controller:
        for i=1 to 1000 do
            Compute the current reference point:
            x_cr=Slerp(x_r,i);
            Send joint torques: ObjImp();
        Open Finger and move back to initial grasp:
        InitialGrasp();
return 0;
```

D. Discussion

During the robust grasping, we didn't consider the problem of grasp stability. As studied in [25], the object dynamic stability will be closely related to the choice of grasp stiffness. In future work, we will investigate ways in which to shape the stiffness while taking the grasp stability into account.

Second, currently the initial grasp and grasp stiffness are predefined in our experiments, which is based on the assumption that the given grasp can realize the desired object impedance. However, given the object impedance specification and a multi-fingered robotic hand, how to choose a grasp that can realize this desired impedance will be a challenging extension direction. One of the possible ways will be extending the optimization framework for grasp synthesis in our previous work [26].

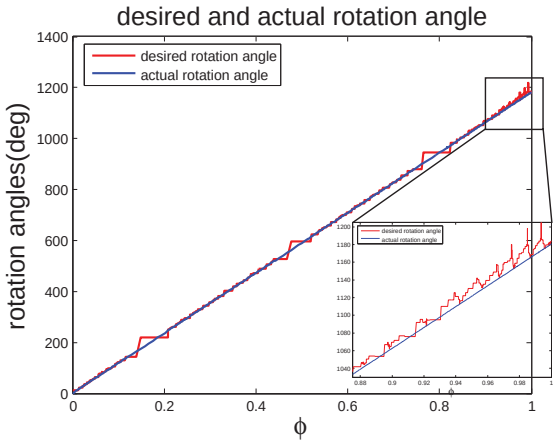
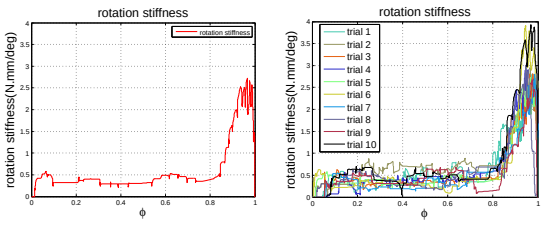


Fig. 11: The learned reference trajectory for trial 5. ϕ is the variable that represents the status of completion of the task, which is chosen as the ratio between current rotational angle and maximal rotational angle.



(a) (b)

Fig. 12: (**a**) The learned desired object stiffness for trial 5. The stiffness will significantly increase at the last phase of bulb replacement. (**b**) The learned desired object stiffness for 10 different trials.

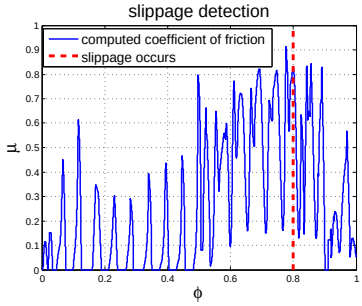
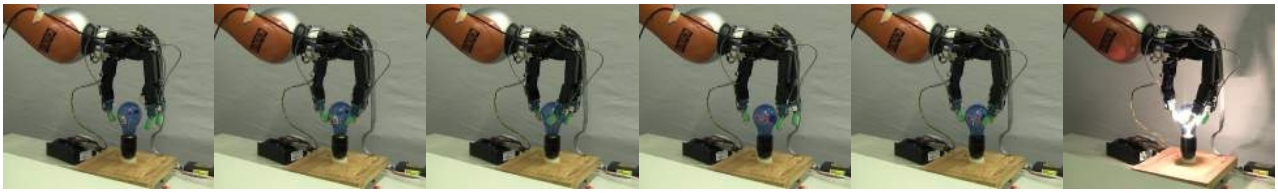


Fig. 14: The computed cofficient of friction on the fingertip of thumb during one successful implementation.



(a) (b) (c) (d) (e) (f)

Fig. 13: The snapshots for dexterous manipulation. The video for this demo is available at: `http://lasa.epfl.ch/~miao/bulb_replace.wmv`

VI. CONCLUSIONS

In this paper, an object-level impedance learning approach was proposed for both robust grasping and dexterous manipulation. For robust grasping, the relative stiffness is specified by measuring the displacement of object under perturbation. For dexterous manipulation, the desired reference trajectory and the desired object impedance is learned through an optimization-based approach. The results show that a varying stiffness is more suitable in our task. Both of these approaches are validated on a multi-fingered robotic hand. We are currently working on integrating tactile feedback into the object impedance controller for grasping stiffness specification.

ACKNOWLEDGMENT

Miao Li was supported by the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement nº 288533 ROBOHOW.COG. Hang Yin was supported partly by a FCT doctoral grant (SFRH/BD/51933/2012) under the IST-EPFL Joint Doctoral Initiative and by the Swiss National Center of Robotics Research. Kenji Tahara was supported by JSPS Grant-in-Aid for Young Scientists (A) (25700028).

REFERENCES

[1] Z. Li, P. Hsu, and S. Sastry, "Grasping and coordinated manipulation by a multifingered robot hand.," *The International Journal of Robotics Research*, vol. 8, no. 4, pp. 33–50, 1989.

[2] T. Yoshikawa and X.-Z. Zheng, "Coordinated dynamic hybrid position/force control for multiple robot manipulators handling one constrained object," *The International Journal of Robotics Research*, vol. 12, no. 3, pp. 219–230, 1993.

[3] Z. Li, Z. Qin, S. Jiang, and L. Han, "Coordinated motion generation and real-time grasping force control for multifingered manipulation," in *Proceedings of International Conference on Robotics and Automation (ICRA), 1998.*

[4] S. A. Schneider and R. H. Cannon, "Object impedance control for cooperative manipulation: theory and experimental results," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 3, pp. 383–394, 1992.

[5] T. Wimbock, C. Ott, and G. Hirzinger, "Analysis and experimental evaluation of the intrinsically passive controller (IPC) for multifingered hands," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2008.*

[6] K. Tahara, S. Arimoto, and M. Yoshida, "Dynamic object manipulation using a virtual frame by a triple soft-fingered robotic hand," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2010.*

[7] T. Wimbck, C. Ott, A. Albu-Schffer, and G. Hirzinger, "Comparison of object-level grasp controllers for dynamic dexterous manipulation," *The International Journal of Robotics Research*, vol. 31, no. 1, pp. 3–23, 2012.

[8] M. T. Mason and J. K. Salisbury, *Robot Hands and the Mechanics of Manipulation*. The MIT series in Artificial Intelligence, Cambridge, Massachusetts: The MIT Press, 1985.

[9] H. Liu and G. Hirzinger, "Cartesian impedance control for the DLR hand," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS), 1999.*

[10] K. Tahara, K. Maruta, A. Kawamura, and M. Yamamoto, "Externally sensorless dynamic regrasping and manipulation by a triple-fingered robotic hand with torsional fingertip joints," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2012.*

[11] N. Hogan, "Impedance control - an approach to manipulation. i - theory. II - implementation. III - applications," *ASME Transactions Journal of Dynamic Systems and Measurement Control B*, vol. 107, pp. 1–24, Mar. 1985.

[12] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer Publishing Company, Incorporated, 1st ed., 2008.

[13] M. Cutkosky and I. Kao, "Computing and controlling compliance of a robotic hand," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 2, pp. 151–165, 1989.

[14] K. Shimoga and A. Goldenberg, "Grasp admittance center: Choosing admittance center parameters," in *American Control Conference, 1991*, pp. 2527–2532, 1991.

[15] B.-H. Kim, B.-J. Yi, S.-R. Oh, and I. H. Suh, "Task-based compliance planning for multifingered hands," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2001.*

[16] B.-H. Kim, B.-J. Yi, S.-R. Oh, and I. H. Suh, "Fundamentals and analysis of compliance characteristics for multifingered hands," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2001.*

[17] B.-H. Yang and H. Asada, "Progressive learning and its application to robot impedance learning," *IEEE Transactions on Neural Networks*, vol. 7, no. 4, pp. 941–952, 1996.

[18] J. Buchli, F. Stulp, E. Theodorou, and S. Schaal, "Learning variable impedance control," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 820–833, 2011.

[19] P. Sikka and B. J. McCarragher, "Stiffness-based understanding and modeling of contact tasks by human demonstration," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS), 1997.*

[20] K. Kronander and A. Billard, "Online learning of varying stiffness through physical human-robot interaction," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2012.*

[21] E. L. Sauser, B. Argall, G. Metta, and A. Billard, "Iterative learning of grasp adaptation through human corrections.," *Robotics and Autonomous Systems*, vol. 60, no. 1, pp. 55–71, 2011.

[22] A. Okamura, N. Smaby, and M. Cutkosky, "An overview of dexterous manipulation," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2000.*

[23] T. Yoshikawa, "Multifingered robot hands: Control for grasping and manipulation," *Annual Reviews in Control*, vol. 34, no. 2, pp. 199 – 208, 2010.

[24] B. Huang, S. El-Khoury, M. Li, J. J. Bryson, and A. Billard, "Learning a real time grasping strategy," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2012.*

[25] C.-H. Xiong, Y.-F. Li, H. Ding, and Y.-L. Xiong, "On the dynamic stability of grasping," *The International Journal of Robotics Research*, vol. 18, no. 9, pp. 951–958, 1999.

[26] S. El Khoury, M. Li, and A. Billard, "Bridging the gap: One shot grasp synthesis approach," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS), 2012.*

Chapter 7

Appendix D

[Full Text] Miao, Li, Yasemin Bekiroglu, Danica Kragic and Billard, A. (2014) Learning of Grasp Adaptation through Experience and Tactile Sensing. International Conference on Intelligent Robots and Systems.

Learning of Grasp Adaptation through Experience and Tactile Sensing

Miao Li[1], Yasemin Bekiroglu[2], Danica Kragic[2] and Aude Billard[1]

Abstract— **To perform robust grasping, a multi-fingered robotic hand should be able to adapt its grasping configuration, i.e., how the object is grasped, to maintain the stability of the grasp. Such a change of grasp configuration is called grasp adaptation and it depends on the controller, the employed sensory feedback and the type of uncertainties inherit to the problem. This paper proposes a grasp adaptation strategy to deal with uncertainties about physical properties of objects, such as the object weight and the friction at the contact points. Based on an object-level impedance controller, a grasp stability estimator is first learned in the object frame. Once a grasp is predicted to be unstable by the stability estimator, a grasp adaptation strategy is triggered according to the similarity between the new grasp and the training examples. Experimental results demonstrate that our method improves the grasping performance on novel objects with different physical properties from those used for training.**

I. INTRODUCTION

Robust grasping is one of the most important capabilities that robots are expected to have. This ability becomes particularly crucial when robots start to work in unstructured environments. Due to the discrepancy between the model and the actual world, or even the errors from the adequate robot control, the planned grasps will hardly be executed without any error. Thus the stability of the final grasp can not be guaranteed even though it has been taken into account in the grasp planing stage. Several works have attempted to integrate different sources of uncertainties in the grasp planing algorithms [1], [2]. But these analytical methods still can not ensure the stability of the final grasps during implementation.

In this paper, we follow another common approach to deal with grasping uncertainty by using sensory feedback in the grasp execution stage. To be more specific, we use the tactile sensing on the fingertips to provide information about the uncertainty of object mass, coefficient of friction, etc. Depending on the tactile sensing, the grasp adaptation mechanism is triggered to increase the probability of achieving a stable grasp. Comparing with other types of sensors, tactile sensors can be more informative to determine information on the physical properties of the grasped object, such as its mass, center of mass, distribution of mass and friction of coefficient. To build estimates of these quantities requires to develop exploratory tactile and manipulation strategies. This paper proposes a method to adapt the grasp to deal with grasping uncertainties from these physical properties.

To cope with uncertainty in grasping, most previous works using sensory feedback focused on uncertainties originating from imprecise model of object's geometry [3], [4], object position and orientation [5], [6], [7], [8]. These geometric uncertainties can directly influence the relative configuration between the robotic hand and the object, upon which the grasp stability is built [9]. In these studies, either the visual feedback or the tactile (force) feedback is used to compensate for the uncertainties by locally adjusting the hand pose or the finger joint angles. The criteria for the adjustment is usually predefined by humans [8] or learned from a large grasp database in simulation [10], [11].

However, as studied in [2], [12], [13], the object physical properties that are usually unknown prior to contact, such as coefficient of friction, object weight and center of mass, can also significantly affect the stability or feasibility of the final grasps. For instance, a robotic hand will likely drop a grasped object when lifting it if the object is heavier than assumed, even though the hand is in the desired configuration. To alleviate the uncertainty of these physical properties of the object, an additional grasping force controller is usually employed to carefully adapt the grasping forces to unknown object weight and friction conditions at the contact points [14]. However, before transiting to force controller, a position controller is still required to correct the geometric uncertainties. The transition between position controller and force controller can result in various different discrete contact states, each of which usually needs to be treated separately [14], [15]. Moreover, it is still very difficult to precisely control the grasping forces on the fingertips in real scenarios due to the uncertainty from finger dynamics and object geometry [16], [17], even with the recent advanced tactile sensors such as BioTac [18].

Instead of regulating the contact forces explicitly, one may use impedance-based controllers [19]. Wimbock *et al.* [20] and Tahara *et al.* [21] developed impedance controller at the object level that prove to be very robust. Both methods rely on the concept of *Virtual Frame* that builds an estimate of the position and orientation of the object solely based on the positions of fingers in contact with the object. Since the virtual frame does not require explicit information on the object's geometry, it can tackle uncertainty in these estimates.

In these works, two control parameters are used for stable grasping: the stiffness K and the rest length L. The stiffness is computed at the object's level and the rest length is the desired distance between each fingertip and the origin of the virtual frame. By varying the stiffness, the grasping forces at

[1]M. Li and A. Billard are with Learning Algorithms and Systems Laboratory (LASA) at École Polytechnique Fédérale de Lausanne (EPFL), Switzeland {miao.li, aude.billard}@epfl.ch

[2] Y. Bekiroglu and D. Kragic are with KTH Royal Institute of Technology, Stockholm, Sweden, as members of the Computer Vision & Active Perception Lab., Centre for Autonomous Systems {yaseminb, danik}@kth.se

each contact point can be regulated, while the grasp configuration can be locally adjusted by changing the rest length. However, these methods rely on a heuristic to choose these two parameters for different objects. Moreover, when the object's physical properties differ from those hypothesized to generate the stable grasp, there is no strategy to guide the controller to adapt these two parameters.

In this paper, we extend the virtual frame approach of [22] and propose an adaptation scheme for the object-level impedance controller using tactile feedback S to change the control parameters K and L so as to maintain a stable grasp. Since the objective of grasp adaptation is to ensure the stability of the final grasps, a grasp stability estimator will be designed to dynamically predict the stability of the current grasp. Compared with previous works on grasp stability estimation [5], [23], our grasp stability estimator is defined in the object frame and thus it is independent of the hand kinematics. The general framework for grasp adaptation is shown in Fig. 1. The two main components of this framework, i.e., stability estimation and grasp adaptation, are highlighted and will be detailed in the following sections.

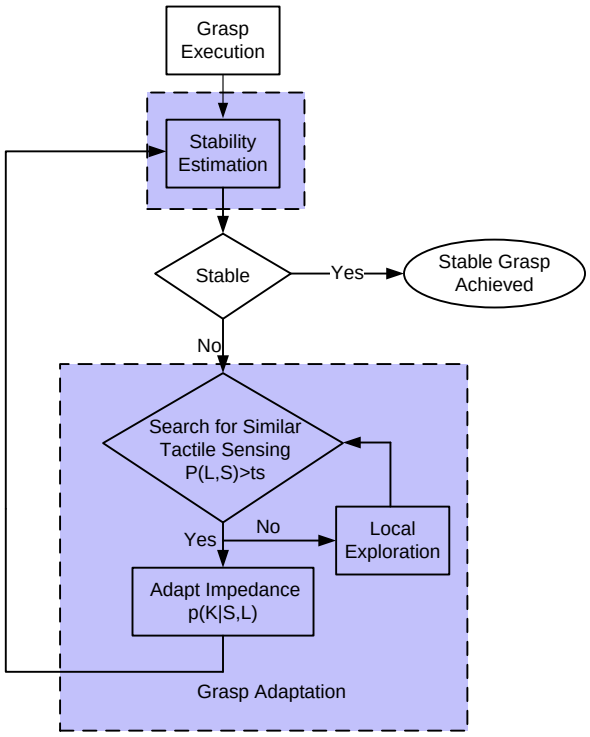


Fig. 1. The pipeline for grasp adaptation using tactile sensing, including two main components: *Grasp Stability Estimation* and *Grasp Adaptation*.

II. Grasp Stability Estimation

In this section, we describe our approach to learn the grasp stability estimator. This approach is similar to the previous methods by Bekiroglu *et al.* [23] and Dang *et al.* [5], but we use a different grasp feature that is directly extracted from the object-level impedance controller [22].

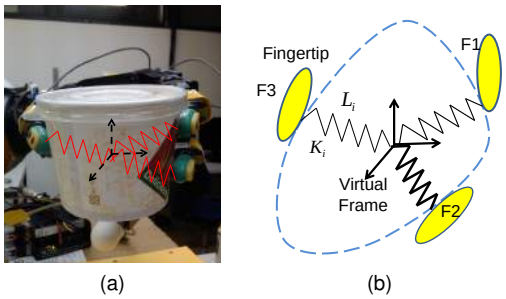


Fig. 2. The scheme for object-level impedance controller. The Virtual Frame (VF) is defined as the center of the three contact points and the controller will control the position and orientation of the VF instead of the real object configuration.

First, we describe the data used for training the grasp stability estimator. In this paper, we only consider precision grasps by three fingers, however our method can be easily extended to grasps using more than three fingers. The notation used is as follows:

- $D = \{(K^i, L^i, S^i)\}_{i=1...N}$ denotes a dataset with N observations.
- $K^i = (K^i_1, K^i_2, K^i_3) \in \mathbb{R}^3$ denotes the grasp stiffness at each fingertip, as shown in Fig. 2.
- $L^i = (L^i_1, L^i_2, L^i_3) \in \mathbb{R}^3$ denotes the rest length at three fingertips, where L^i_j is the distance from the j-th fingertip to the center of virtual frame.
- $S^i = (S^i_1, S^i_2, S^i_3) \in \mathbb{R}^{57}$ denotes the tactile reading at three fingertips, where $S^i_j \in \mathbb{R}^{19}$ is the tactile reading from the j-th fingertip.

The recorded data thus consist of tactile readings S and parameters of the object-level controller, grasp stiffness K and rest length L. The tactile readings are high dimensional and redundant. Thus we use the Principal Component Analysis (PCA) to reduce the dimensionality. Hereafter, we will use S to denote the tactile reading after dimensionality reduction. Before the training procedure, all the data are normalized to zero mean with range $[-1, 1]$.

With the recorded data, we formulate the grasp stability estimation as a *one-class classification* problem. This means that only the positive data, i.e, data from stable grasps, are used to learn the boundary of the stable grasp region. This is also different from previous approaches [5], [23], where both positive and negative data are used to train the model. In general, the unstable grasps in our experiments can be seen as the noisy versions of the stable grasps, i.e, too many ways to perform a bad grasp, because they do not have inherit structures to learn. Due to this, binary classification experiments did not improve the classification accuracy. Therefore it is sufficient to use only one-class approach for our data. Moreover, we noticed that in practice, it may require a huge amount of time and expense (damaging the object or the finger) to collect sufficient negative data, which would make the approach infeasible in practice. Therefore,

in this work, we only use the positive data to learn the region of stable grasps.

In order to learn the boundary of the stable grasp region, two types of nonlinear classifiers, namely Gaussian Mixture Model (GMM) and Support Vector Machine (SVM) are used. GMM is a generative approach that models the probability distribution over data. The likelihood of a grasp $X_* = (K_*, L_*, S_*)$ under a GMM model denote by Ω with m Gaussian components is given by:

$$p(X_*|\Omega) = \sum_{i=1}^{m} \pi_i \mathcal{N}(X_*|\mu_i, \Sigma_i) \tag{1}$$

where π_i is the prior of the ith Gaussian component and $\mathcal{N}(\mu_i, \Sigma_i)$ is the Gaussian distribution with mean μ_i and covariance Σ_i as follows:

$$\mu_i = \begin{bmatrix} \mu_{K,i} \\ \mu_{L,i} \\ \mu_{S,i} \end{bmatrix}, \Sigma_i = \begin{bmatrix} \Sigma_{KK,i} & \Sigma_{KL,i} & \Sigma_{KS,i} \\ \Sigma_{LK,i} & \Sigma_{LL,i} & \Sigma_{LS,i} \\ \Sigma_{SK,i} & \Sigma_{SL,i} & \Sigma_{SS,i} \end{bmatrix} \tag{2}$$

A new grasp is said to be stable if its likelihood of being generated by the model is greater than a fixed threshold, i.e. $p(X_*) > te$. The discrimination threshold $te \in [a, b]$ is chosen according to the ROC (Receiver Operating Characteristic) curve, where the bounds a and b correspond to the minimal and maximal likelihood of each Gaussian component at two standard deviation, respectively, that is,

$$\begin{aligned} Lik_{2\sigma}(i) &= (2\pi)^{-\frac{d}{2}} |\Sigma_i|^{-\frac{1}{2}} e^{-2} \\ a &= \min_{i=1\ldots m} Lik_{2\sigma}(i) \\ b &= \max_{i=1\ldots m} Lik_{2\sigma}(i) \end{aligned} \tag{3}$$

As a comparison, we also applied the SVM classification to our problem. SVM for one class classification is a maximum margin classifier that attempts to separate the data from the origin with maximum margin in the feature space [24]. Here, we use SVM implementation from [25].

It is worth mentioning that other one-class classification methods [26], can also be adopted here for the grasp stability estimation. In our work, GMM is chosen because (1) it has already shown its ability at estimating region of stable grasps in the high-dimensional joint space for different hands [27] [28], (2) the grasp adaptation strategy can be naturally derived from the learned model, as will be explained in the next section.

III. Grasp Adaptation

The grasp adaptation procedure is required when the current grasp is predicted to be unstable. In this case, corrective actions should be launched and driven by the current tactile information. In our controller, grasp adaptation consists of changing the object level impedance controllers according to the tactile feedback. Specifically, we will adapt the grasp stiffness K and the rest length L. The goal of grasp adaptation is to find a similar stable grasp using our model constructed in Section II. The following parts of this section will present two corrective actions according to the similarities between the acquired tactile readings and the training dataset.

A. Impedance Adaptation

As illustrated in Fig. 1, when a grasp $X = (K, L, S)$ is predicted to be unstable, our first adaptation strategy is to adapt the grasp stiffness. This regulates indirectly the contact forces at each finger. This adaptation strategy corresponds to a typical response in humans whereby the grip force is increased to maintain the stability of grasped (or manipulated) object [29]. Once an unstable grasp is detected, such as slippage occurring on one fingertip, the desired grasp stiffness $\hat{K}$ is predicted from the GMM through regression, given the current rest length L and tactile reading S.

During the execution of grasp adaptation, however, we first need to check whether the current query point $q = [L^T, S^T]^T$ is likely enough with respect to learned model. In other words, we need to check if the current query point q is close enough to the training examples. This step is required mainly because a query point with low likelihood, i.e., far away from the training examples, may give a very poor prediction that may lead to even more unstable grasps.

In order to determine if the current query point q is likely under the learned model Ω, we compute the Mahalanobis distance from q to the center of each Gaussian component. The distance to the ith component is defined as:

$$f_i(q, \Omega) = \frac{1}{2}(q - \mu_{q,i})^T \Sigma_{q,i}^{-1} (q - \mu_{q,i}) \tag{4}$$

where $i = 1, ..., m$ is the index of Gaussian components, $\mu_{q,i}$ and $\Sigma_{q,i}$ are the corresponding components in (1) as follows:

$$\mu_{q,i} = \begin{bmatrix} \mu_{L,i} \\ \mu_{S,i} \end{bmatrix}, \Sigma_{q,i} = \begin{bmatrix} \Sigma_{LL,i} & \Sigma_{LS,i} \\ \Sigma_{SL,i} & \Sigma_{SS,i} \end{bmatrix} \tag{5}$$

Note that compared to the marginal likelihood $p(q|\Omega)$, the Mahalanobis distance $f_i(q, \Omega)$ has the advantage to give the same importance to each Gaussian component, due to the absence of the prior in (4) and the fact that we normalize the distance to the center of the Gaussian. This has also the merit to avoid biasing the selection of grasps toward large Gaussian component, which may happen because of the non-uniform distribution of the training dataset.

In this work, we consider a query point q is likely enough to belong to the learned model, if its Mahalanobis distance to at least one of the Gaussians of the model is below two, i.e., $\exists i, i = 1, \ldots, m, f_i(q, \Omega) < 2$. In other words, if the query point is inside the two standard deviations of any Gaussians of the model, then it is considered to be close enough to the learned model.

When the current query point q is likely enough under the model, the expected grasp stiffness is obtained by taking the expectation over the conditional distribution, $p(\hat{K}|L, S, \Omega)$ [30], which can be computed as follows:

$$E\{p(\hat{K}|L, S, \Omega)\} = \sum_{i=1}^{m} h_i(\mu_{k,i} + \Sigma_{Kq,i}\Sigma_{q,i}^{-1}(q - \mu_{q,i})) \tag{6}$$

where $\Sigma_{Kq,i} = \begin{bmatrix} \Sigma_{KL,i} \\ \Sigma_{KS,i} \end{bmatrix}$, and $h_i = \frac{\pi_i \mathcal{N}(q|\mu_{q,i}, \Sigma_{q,i})}{\sum_{j=1}^{m} \pi_j \mathcal{N}(q|\mu_{q,j}, \Sigma_{q,j})}$.

The expected grasp stiffness value will be thus set directly to the object level impedance controller, as shown in Fig. 1.

B. Local Exploration

When the current query point q is far away from all the training examples, a second adaptation strategy takes place to project the current query point to the closest point q_* in the training dataset. Since in the controller only the rest length is required to determine the grasp configuration, we move the current grasp configuration towards the closest center of the Gaussian components, which is chosen according to the same distance metric defined above (4). The rest length of the closest center of Gaussian component is denoted as $L^c = [L_1^c, L_2^c, L_3^c]^T$.

However, differently from the grasp stiffness, we can not simply change these parameters in the controller since the definition of the virtual frame depends on the contact position at each fingertip. Moreover, the kinematics of each finger and the local geometry of the object surface will also impose several constraints on the possible movements of each fingertip. We therefore adopt an iterative approach by defining the following objective function:

$$J = \sum_{i=1}^{3} (L_i - L_i^c)^2 \tag{7}$$

where $L = [L_1, L_2, L_3]^T$ is the current rest length. In this work, we only consider the adaptation motion from one of the fingers, denoted here as finger 1, see Fig. 3. For a more general case with multi-fingered grasp (more than three fingers), a more elaborate strategy that considers each finger's workspace and the grasp stability would be required to determine which finger should be used for adaptation.

In order to minimize the objective function, the gradient of the objective function with respect to the position of finger 1, i.e., $P_1 = [X_1, Y_1, Z_1]$, is given as $v = [\frac{\partial J}{\partial X_1}, \frac{\partial J}{\partial Y_1}, \frac{\partial J}{\partial Z_1}]^T$, where each component of v is computed as:

$$\frac{\partial J}{\partial X_1} = \sum_{i=1}^{3} \frac{\partial J}{\partial L_i} \frac{\partial L_i}{\partial X_1} \tag{8}$$

$\frac{\partial J}{\partial L_i}$ and $\frac{\partial L_i}{\partial X_1}$ can be easily computed from the definition of J in (7) and the definition of L_i as given in [22].

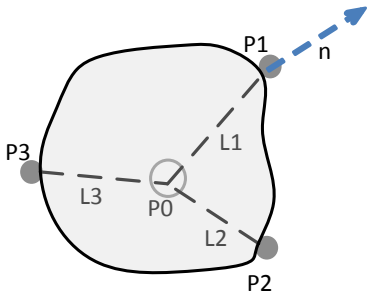


Fig. 3. The scheme for local object exploration. Only finger 1 is used here for local exploration and the normal direction obtained from tactile sensing is also used for guiding the exploration directions.

Due to the constraint of the object surface, the fingertip 1 can not penetrate inside the object surface. Therefore we need to project the gradient onto the tangential surface at fingertip 1: $v^* = v - \langle v, n \rangle n$, where n is the normal direction at fingertip 1. Then the next desired position of finger 1 is given by: $P_1^* = P_1 - \alpha v^*$, where $\alpha \in (0, 1)$ is the step size and is manually set as 0.03 in our experiment[1]. In order to move finger 1 to the desired position, we implement a fingertip impedance controller for finger 1, which is superimposed on the object-level impedance controller.

It is worth noticing that during the local exploration, only the rest length L can be controlled. This does not guarantee that a similar query point q, consisting of the rest length L and the tactile sensing S, can be always found. To this end, a maximum number of steps for local exploration is set in our controller, i.e., $N_{max} = 5$. If a similar query point cannot be found within N_{max} times steps, our controller will recompute the closest center of the Gaussian components according to (4) and repeat the local exploration loop.

IV. Experimental Evaluation

In this section, we demonstrate the effectiveness of our approach on several everyday objects. We first present the experimental setup and the data collection procedure, then present the results on grasp stability estimation and grasp adaptation.

A. Setup and Data Collection

As shown in Fig. 4, we use a 16 DOFs (degree of freedom) *Allegro Hand* from Simlab[2] with four fingers. Each finger has four independent torque-controlled joints. In our experiments we only use three of these four fingers. Each fingertip has been equipped with the tactile sensor *BioTac* from SynTouch[3]. BioTac can provide multi-modal tactile information, such as vibration, temperature and pressure. Here, we only use the 19 dimensional pressure data from the electrode impedances, which are related to the contact features such as contact force, contact location and deformation. The sampling rate is 100Hz in our experiments.

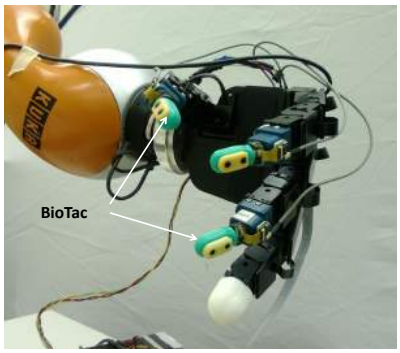


Fig. 4. Allegro Hand equipped with BioTac on the fingertips.

[1]Note that there is a trade-off here. If α is small, the object may fall before finding a stable grasp. If α is too large, the finger may overshoot the desired position or move out of the object surface.

[2]http://www.simlab.co.kr/Allegro-Hand.htm

[3]http://www.syntouchllc.com/

(a) cola can (b) food box (c) box (d) cup

Fig. 5. The four objects used in the experiments for collecting training dataset.

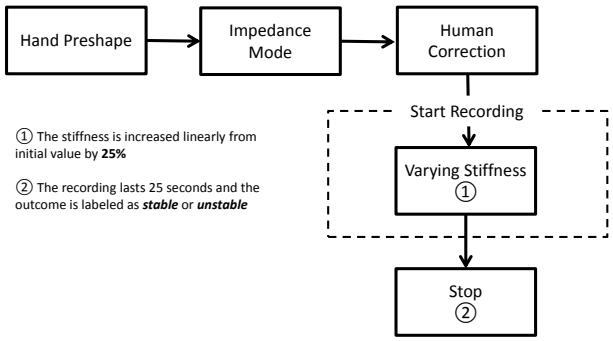


Fig. 6. The data collection procedure. Only one hand preshape is used in our experiment, as shown in Fig. 4. The object is put inside the hand and the control mode is changed to impedance mode. One or more fingertips' position is locally adjusted to set an initial stable grasp. This step is also required since we want to vary the rest length for each trial.

Our primary goal in this paper is to deal with physical uncertainties and therefore we assume that the object is already grasped with a given preshape, Fig. 4. The geometric information about the object is not known a-priori. Four different objects are used in our experiments, shown in Fig. 5. For each object we have five different weights by filling them with different amount of pepper, see Table I. The initial grasp stiffness is manually chosen as a minimal value that can grasp the object in a stable way. Before recording, the initial grasp may be slightly adjusted by the human to change the grasp configuration, i.e., to change the possible value of the rest lengths. Once recording has started, the stiffness is increased linearly from the initial value to 125% of the initial value with 10% random noise. Both the change of initial grasp configuration and the increase of grasp stiffness in the experiments are aiming at increasing the variety of our collected data.

TABLE I

THE WEIGHTS OF DIFFERENT OBJECTS DURING TRAINING

object	object weight(g)				
cola can	16	38	50	66	80
food box	10	26	52	68	88
box	34	56	76	99	121
cup	10	49	98	135	155

The recording procedure lasts 25 seconds for each trial and the whole trial is repeated 5 times for each object weight. Fig. 6 provides an outline of the data collection. In total, we collected $4 \times 5 \times 5 \times 25 \times 100 = 250000$ positive examples. For each object, we also collected several negative examples by either setting the initial stiffness to a very small value or filling much more pepper into the objects. In total, we have 37500 negative examples.

B. Results for Grasp Stability Estimation

The dataset is divided into training and test sets. For the first three objects (cola can, food box, box), one weight is selected randomly and the data collected with this weight is used as the test set. For the cup, all the collected data are grouped into test set. All the negative datapoints are also used for testing. The tactile data has originally $19 \times 3 = 57$ dimensions and the dimensionality is reduced to 8 dimensions based on PCA, hence in total the data has $8 + 3 + 3 = 14$ dimensions.

For training the model defined by (1), the number of Gaussian components, m, is selected using *Bayesian Information Criteria* (BIC) and set to 16 in our experiments, $m = 16$. The other parameters in GMM are learned using the EM algorithm. Figure 7 shows the learned stable grasp region projected onto the first two principal components of the tactile reading. Examples of stable grasps and unstable grasps are given for each object. As shown in Fig. 8, in these projected lower dimensions, some unstable grasps can be easily separated, such as unstable grasp 2, (grasp of food box). This grasp is unstable because one finger (finger 2) loses contact with the object, see the right upper picture in Fig. 7. But in most other cases, when the object starts to slip or tilt slowly from the hand, these unstable grasps cannot be easily separated from stable grasps in the projected lower dimensions. We need to use all the dimensions to compute the marginal likelihood according to (1) and compare it with the threshold te.

In order to select the threshold te in (1), we vary the value of te in (3) with $a = -708.4$ and $b = -35.30$ (The logarithmic likelihood is used here). The ROC curve for the test dataset is shown in Fig. 9. The variance on the curve is calculated based on 5 fold cross-validation.

For our application, false positives should be avoided at all costs, as they correspond to the cases where an unstable grasp is mistaken for a stable grasp. For this reason, we chose a very high te value which corresponds to $TPR = 82.27\%$ and $FPR = 15.01\%$. This result is comparable with the results obtained in [23] for unknown objects.

For comparison, the SVM for one-class classification is also used to predict the grasp stability. We use the RBF (radial basis function) kernel and the parameters are selected using multi-scale grid search. The best performance SVM can obtain is: $TPR = 84.66\%$ and $FPR = 29.72\%$. Although the true positive rates are similar, the SVM has a higher false positive rate. This can be explained by the fact that one-class SVM only tries to separate the training data with origin in the feature space [26], which is a considerable loose constraint compared with a GMM that attempts to model the density of the stable grasp region.

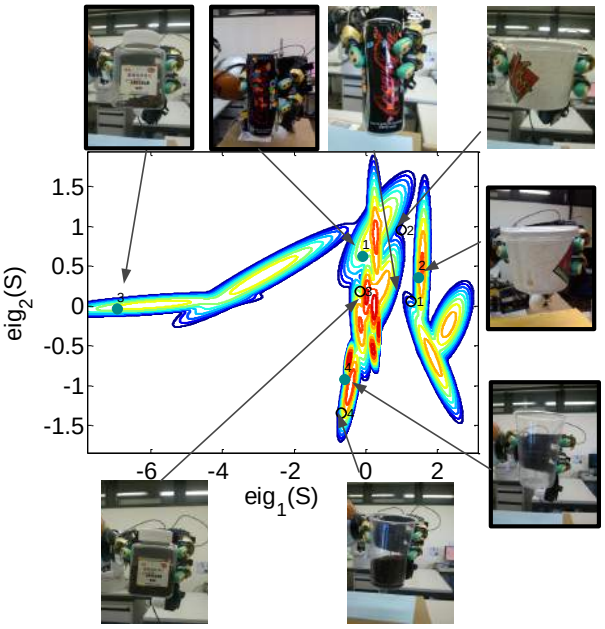
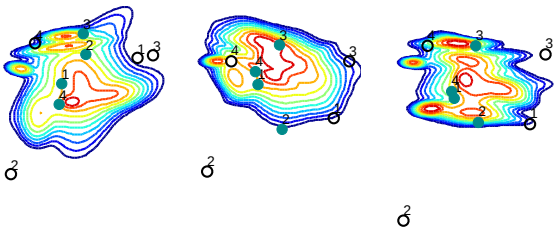


Fig. 7. The density distribution of the learned GMM model. The axes correspond to the projection of the tactile sensing S on the first two principal components. Contours correspond to the isocurve with constant marginal likelihood value $p(S|\Omega)$. Solid dots denote the stable grasps and circles denote the unstable grasps.



(a) $\text{eig}_7(S)$ vs L_1 (b) $\text{eig}_7(S)$ vs L_2 (c) $\text{eig}_7(S)$ vs L_3

Fig. 8. The density distribution of the learned GMM model projected into lower dimensions. The X-axis corresponds to the 7th dimension of the tactile readings and the Y-axis corresponds to the dimension of L_1, L_2 and L_3, respectively. Contours correspond to the isocurve with constant marginal likelihood value $p(S|\Omega)$. Solid dots denote the stable grasps and circles denote the unstable grasps.

C. Results for Grasp Adaptation

To test the validity of our grasp adaptation strategy, first we only use the collected negative datapoints to demonstrate that these grasps can be predicted to be unstable grasps. Also, depending on their similarities with the training dataset, different adaptation strategies should be able to react. For all the 37500 negative datapoints, only 8.70% of them are misclassified as stable grasps. 33.70% of them require the impedance adaptation while 57.60% of them require the local exploration.

We also tested our grasp adaptation approach on the real robotic hand with different objects, see Fig. 10-15. When the object weights are changing by adding pepper or disturbed by

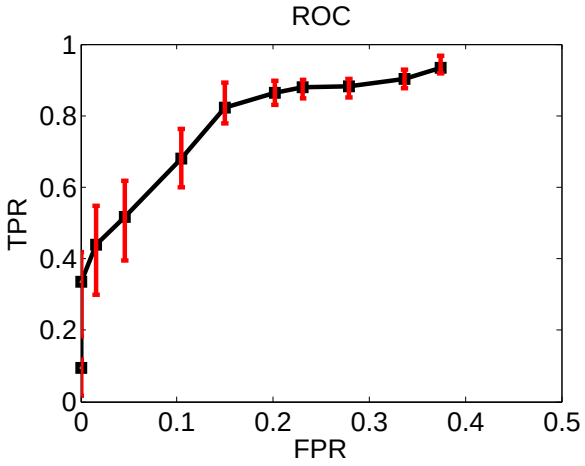


Fig. 9. ROC curve to select the threshold te.

a human[4], the grasp adaptation strategy is triggered to keep the grasp stable, either by varying the grasp stiffness or by changing the location of finger 1. Taking the test of the cup as an example, see Fig. 11, with the increase of the weight, first the stiffness is increasing until the distance computed from (4) is larger than 2. At the second stage (shown as red dots), the finger 1 starts to explore local area until the query point is close again. Finally, the stiffness is changed again to find a stable grasp.

(a) (b) (c) (d)

Fig. 10. A snapshot for the experiment on the cup. **(a)** The initial grasp stiffness is set manually. **(b)** Pepper was poured into the cup to change the cup's weight. **(c)** Finger 1 is changing its location for local exploration. (d) The new grasp configuration after the exploration of finger 1.

To quantify the results of our grasp adaptation strategy, we compare the maximal object weights that the grasp can support with and without the grasp adaptation strategy, see Table II. We use the same initial grasp configuration and grasp stiffness for each object as in the data collection procedure, i.e., the setup for data collection in the first column of Table I. The object weight is still varying by adding pepper and the object is considered as unstable once there is a steady noticeable slippage between the fingertips and the object. Each object is tested five times, both with grasp adaptation and without grasp adaptation.

In Table II, each row corresponds to the average value of the maximal object weight (grams) that the grasp can support for each object, with (with) and without (w/o) grasp adaptation. The standard deviation during the five experiments is also computed and given as $mean \pm std$. The comparison of the results shows that the grasp adaptation

[4]We use human perturbation to simulate the change of the object weight.

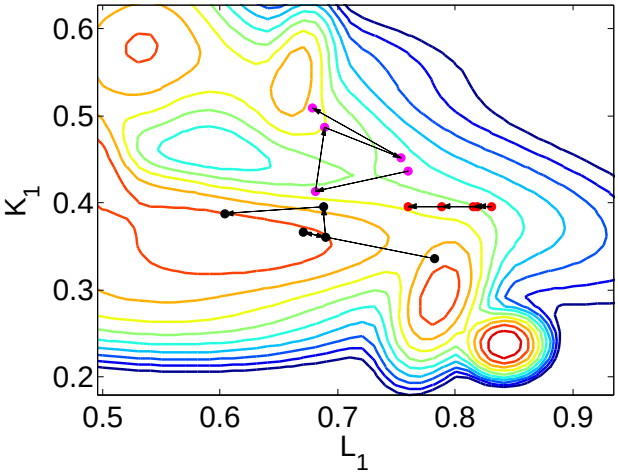


Fig. 11. The joint density distribution of the dimensions L_1 and K_1; The black dots correspond to the impedance adaptation stage for the testing of the cup (Fig. 10(b)). Red dots correspond to the local exploration stage and magenta dots correspond to the impedance adaptation after local exploration. For each stage, we only plot 5 datapoints.

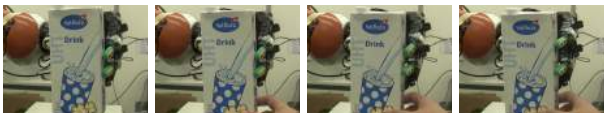


Fig. 12. A snapshot for the experiment on the milkbox. A human is pulling the milkbox downwards **(b)** and at the time shown in **(c)**, finger 1 starts to change its location in order to keep the stability of the grasp.

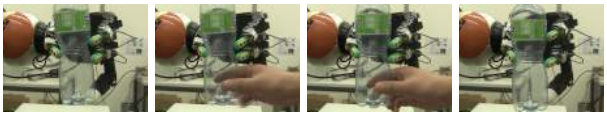


Fig. 13. A snapshot for the experiment on the water bottle.

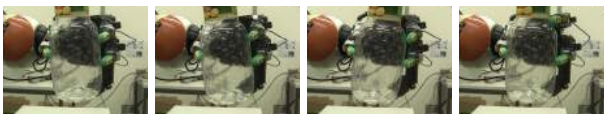


Fig. 14. A snapshot for the experiment on the juice bottle. A human is trying to push the bottle downwards from the top of the bottle **(b)** and finger 1 starts to adapt its location at the moment shown in **(c)**.

Fig. 15. A snapshot for the experiment on the cola bottle.

can have a significant improvement on the maximal weight that a grasp can support.

TABLE II

THE COMPARISON OF THE SUPPORTED OBJECT WEIGHTS (WITH VS. W/O GRASP ADAPTATION)

obj.	cola can	food box	box	cup
w/o	17.2 ± 1.92	12.8 ± 0.84	37.2 ± 2.59	15.0 ± 2.55
with	69.0 ± 6.52	84.0 ± 3.80	121.2 ± 9.20	146.4 ± 5.46

D. Discussion

In our controller, since the virtual frame does not assume prior information about the object's shape, this may lead to problems for local exploration. The exploring finger 1 may move out of the object surface after the local exploration, although this problem can be avoided by detecting the loss of contact and moving the finger back to the previous contact position. Also, instead of using only finger 1, other fingers can also be used for local exploration, which will be especially useful when the object is grasped by more than 3 fingers. This requires a more complicated exploration strategy as well as a grasp planning strategy that takes each finger's adaptability into account during the planning stage [31].

Another issue is about the implementation of local exploration. At present, to do the local exploration, we only implemented a fingertip impedance controller to move finger 1 to the desired position. However, we found that in practice it is not so trivial to choose the proper parameters for the fingertip impedance controller. As shown in Fig. 10 and Fig. 14, the finger 1 may "jump" to the desired position, which sometimes leads to unstable grasps due to either the loss of contact during jump or the large impact during contact. It is more reasonable to implement a contour following controller that can move the exploring finger (finger 1) to the desired position by following the object's contour. However, since the object is grasped and supported by the other fingers during the exploration, the force that the exploring finger applies on the surface should be adapted to the stiffness of the grasped object.

V. CONCLUSION

This paper presented a new framework for grasp adaptation under physical uncertainties of the grasped objects. The adaptation strategy is derived from an object-level impedance controller and learned with training data that is generated using a real robotic hand. We first formulated the grasp stability estimation as a one-class classification problem. A Gaussian Mixture Model is used to model the region of the stable grasps. During the grasp execution, if a grasp is predicted to be unstable, two different adaptation strategies, i.e., impedance adaptation and local exploration, are selectively triggered according to the similarity between the current unstable grasp and the examples in the stable grasp regions. The effectiveness of our approach is validated on the Allegro hand equipped with BioTac tactile sensors for the experiments.

ACKNOWLEDGMENT

This work was supported by the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement n^o 288533 ROBOHOW.COG.

REFERENCES

[1] R. C. Brost, "Planning robot grasping motions in the presence of uncertainty," Tech. Rep. CMU-RI-TR-85-12, Robotics Institute, Pittsburgh, PA, July 1985.

[2] Y. Zheng and W.-H. Qian, "Coping with the grasping uncertainties in force-closure analysis.," *International Joournal of Robotic Research*, vol. 24, no. 4, pp. 311–327, 2005.

[3] B. Kehoe, D. Berenson, and K. Goldberg, "Toward cloud-based grasping with uncertainty in shape: Estimating lower bounds on achieving force closure with zero-slip push grasps," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2012.*

[4] V. N. Christopoulos and P. R. Schrater, "Handling shape and contact location uncertainty in grasping two-dimensional planar objects," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS), 2007.*

[5] H. Dang and P. Allen, "Stable grasping under pose uncertainty using tactile feedback," *Autonomous Robots*, pp. 1–22, 2013.

[6] J. Laaksonen, E. Nikandrova, and V. Kyrki, "Probabilistic sensor-based grasping," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[7] K. Hsiao, S. Chitta, M. Ciocarlie, and E. Jones, "Contact-reactive grasping of objects with partial shape information," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS), 2010.*

[8] J. Felip and A. Morales, "Robust sensor-based grasp primitive for a three-finger robot hand," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS), 2009.*

[9] J. Kim, K. Iwamoto, J. Kuffner, Y. Ota, and N. Pollard, "Physically based grasp quality evaluation under pose uncertainty," *IEEE Transactions on Robotics*, vol. 29, no. 6, pp. 1424–1439, 2013.

[10] H. Dang and P. Allen, "Grasp adjustment on novel objects using tactile experience from similar local geometry," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[11] J. Steffen, R. Haschke, and H. Ritter, "Experience-based and tactile-driven dynamic grasp control," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS)*, 2007.

[12] S. El-Khoury, M. Li, and A. Billard, "Bridging the gap: One shot grasp synthesis approach," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS), 2012.*

[13] S. El-Khoury, M. Li, and A. Billard, "On the generation of a variety of grasps," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1335–1349, 2013.

[14] J. M. Romano, K. Hsiao, G. Niemeyer, S. Chitta, and K. J. Kuchenbecker, "Human-inspired robotic grasp control with tactile sensing.," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1067–1079, 2011.

[15] T. Schlegl, F. Freyberger, S. Haidacher, F. Pfeiffer, M. Buss, and G. Schmidt, "Compensation of discrete contact state errors in regrasping experiments with the tum-hand," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS), 1999.*

[16] K.-C. Nguyen and V. Perdereau, "Fingertip force control based on max torque adjustment for dexterous manipulation of an anthropomorphic hand," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[17] W. Yang, M. Li, and X. Zhang, "Robust robotic grasping force optimization with uncertainty.," in *ICIRA*, vol. 6425 of *Lecture Notes in Computer Science*, pp. 264–275, Springer, 2010.

[18] N. Wettels, V. J. Santos, R. S. Johansson, and G. E. Loeb, "Biomimetic tactile sensor array.," *Advanced Robotics*, vol. 22, no. 8, pp. 829–849, 2008.

[19] T. Yoshikawa, "Multifingered robot hands: Control for grasping and manipulation," *Annual Reviews in Control*, vol. 34, no. 2, pp. 199 – 208, 2010.

[20] T. Wimbock, C. Ott, and G. Hirzinger, "Analysis and experimental evaluation of the intrinsically passive controller (IPC) for multifingered hands," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2008.*

[21] K. Tahara, S. Arimoto, and M. Yoshida, "Dynamic object manipulation using a virtual frame by a triple soft-fingered robotic hand," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2010.*

[22] M. Li, H. Yin, K. Tahara, and A. Billard, "Learning object-level impedance control for robust grasping and dexterous manipulation," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2014.*

[23] Y. Bekiroglu, J. Laaksonen, J. A. Jorgensen, V. Kyrki, and D. Kragic, "Assessing grasp stability based on learning and haptic data," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 616–629, 2011.

[24] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, pp. 1443–1471, July 2001.

[25] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[26] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, pp. 15:1–15:58, July 2009.

[27] B. Huang, S. El-Khoury, M. Li, J. J. Bryson, and A. Billard, "Learning a real time grasping strategy.," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2013.*

[28] E. L. Sauser, B. Argall, G. Metta, and A. Billard, "Iterative learning of grasp adaptation through human corrections.," *Robotics and Autonomous Systems*, vol. 60, no. 1, pp. 55–71, 2012.

[29] R. S. Johansson, "Sensory input and control of grip," in *Novartis Foundation 218: Sensory Guidance of Movement*, pp. 45–63, 1998.

[30] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, "Active learning with statistical models.," *Journal of Artificial Intelligence Research*, vol. 4, pp. 129–145, 1996.

[31] K. Hang, J. A. Stork, and D. Kragic, "Hierarchical fingertip space for multi-fingered precision grasping," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS)*, 2014.

Chapter 8

Appendix E

[Full Text] Kaiyu Hang, Miao Li, Johannes A. Stork, Yasemin Bekiroglu, Florian T. Pokorny, Aude Billard and Danica Kragic (2014) Hierarchical Fingertip Space: From Grasp Planning to In-Hand Adaptive Grasping.

Hierarchical Fingertip Space: From Grasp Planning to In-Hand Adaptive Grasping

Kaiyu Hang, Miao Li, Johannes A. Stork, Yasemin Bekiroglu, Florian T. Pokorny, Aude Billard and Danica Kragic

***Abstract*—Grasp planning and execution are two major components of robotic grasping systems. Although rapid developments have been made in both areas in the past decade, they are still developed mostly in isolation. This paper presents a system that couples grasp planning, tactile sensing and adaptive control in a single framework, with the focus of improving the system robustness by closing the loop between planning and control to approach disturbance rejection in fingertip grasping. During the planning phase, our system generates grasps in the *Hierarchical Fingertip Space* by optimizing both the grasp stability and the adaptability. Once a grasp is executed, our system feeds tactile readings and grasp rest length back to a probabilistic model learned over both tactile readings and hand configurations, and estimates its stability status online. When a grasp is going to be unstable, our system remains the stability by either adapting the grasp force or relocating fingertips, which is planned in the *Hierarchical Fingertip Space* by fast graph exploration and pruning. The system is implemented on an Allegro hand mounted on a Kuka LWR arm, we present a broad set of experimental evaluations and show that this system improves the robustness of fingertip grasping, and that it achieves the performance that can not be gained by either subsystems individually.**

***Index Terms*—Fingertip grasping, Hierarchical Fingertip Space, grasp optimization, grasp adaptation**

Fig. 1. Adaptable fingertip grasping in the *Hierarchical Fingertip Space*.

I. Introduction

OWING to the complexity of fingertip grasping, subproblems such as grasp stability estimation [1], contact-level grasp synthesis [2], hand configuration calculation [3], hand dynamic control [4] and grasp adaptation [5] etc. have been separately addressed to tackle either planning or control in grasping systems. However, since robotic grasping is a composite system that involves various components, addressing it on individual aspects can usually raise limits [6], [7]. In the research of grasp synthesis, which deals with the planning part of grasping systems, stability and task related affordability are usually the main focuses. However, problems such as preparing a grasp for future interactions with the environment [8], [9], e.g., external perturbations, is rarely considered. On the other hand, the control of grasp execution has been usually focusing on how to keep the stability of grasps using available online feedback [10], [11]. The rich information used during planning phase, such as object's shape and hand's properties that are very useful for online re-planning, is usually ignored. This separated way of considering planning and control makes the subsystems do not benefit from each other and, more importantly, neglects some problems by assuming the other will solve them, e.g., the controller will execute exactly what is planned by the planner.

In the research that relates human grasping to robotic grasping [12], many works imply that humans tend to adopt a two phases strategy for grasping: First, visual perception and experiences are used to plan grasps, and then the grasp execution is dominated by tactile sensing with assistance of human's prior knowledge of the object. Since humans have knowledge of what grasps are more comfortable and reliable for their hands, they plan grasps with preferences of contact locations and hand configurations. Furthermore, humans execute planned grasps adaptively to what they feel in real-time, so that the execution is more robust against uncertainties, e.g., external perturbations or positioning errors, which are not predictable by planners. Following this direction, and based on our previous works of *Hierarchical Fingertip Space*[1] based grasp synthesis [13] and tactile sensing based grasp adaptation [5], we develop a fingertip grasping system that couples grasp planning, tactile

K. Hang, J. A. Stork, aY. Bekiroglu, F. T. Pokorny and D. Kragic are with KTH Royal Institute of Technology, Stockholm, Sweden, as members of the Computer Vision & Active Perception Lab., Centre for Autonomous Systems {kaiyuh, jastork, yaseminb, fpokorny, dani}@kth.se.

M. Li and A. Billard are with Learning Algorithms and Systems Laboratory (LASA) at École Polytechnique Fédérale de Lausanne (EPFL), Switzerland {miao.li, aude.billard}@epfl.ch.

[1]In the rest of this paper, the *Hierarchical Fingertip Space* is abbreviated as *HFTS*.

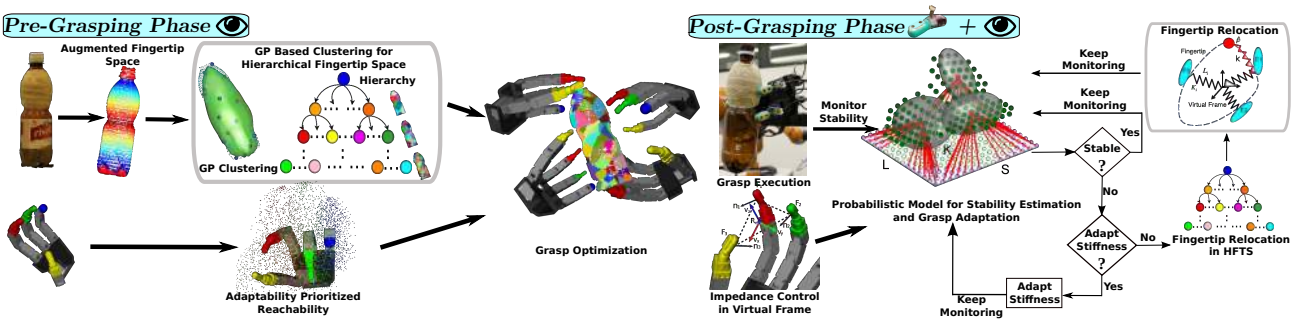


Fig. 2. Schematic overview of the proposed system.

sensing and adaptive control to consider grasp synthesis and robust execution in a single framework. This system operates in two phases: 1) the *pre-grasping* phase where the system synthesizes grasps in the *HFTS*, and 2) the *post-grasping* phase during which the system executes, monitors and adapts grasps in terms of its tactile readings and learned experiences, so as to realize the planned grasp while adapting it according to the online feedback to approach disturbance rejection. Note that when we represent the discrepancies of a physical system and our knowledge of it, disturbances and uncertainties are synonymous [14], we will hereafter refer to these two terms as the same concept. Before getting into the details of the system, we give a brief overview of the proposed fingertip grasping system by defining the two phases of it, as well as explaining our strategies for approaching disturbance rejection.

Pre-grasping: The pre-grasping phase of our system is defined as the period in which the system does grasp planning with only visual sensory information, such as object point cloud used in this work. Since the planned grasp will be executed by the controller afterwards, we hence expect the grasp to have certain properties preferred by the controller. For this, we consider the quality of a grasp in terms of not only the stability, but also the qualities of its contact locations and the corresponding hand configurations. Basically, as shown in Fig. 2, the grasp planning in our system is conducted in 3 steps in a sequence: 1) given the preference of the controller, a discrete set of viable contact locations – fingertip space – is extracted and augmented from the object point cloud, and 2) the system builds a structured representation of the fingertip space – *HFTS* – to recursively abstract the fingertip space into a hierarchy of multi-resolution spaces, and then 3) by formulating grasp synthesis as a combinatorial optimization problem in the *HFTS*, our system generates grasps by multilevel refinement of grasp stability, adaptability and reachability, and produces contact locations together with hand configurations for the controller to execute. This phase is detailed in Sec. III.

Post-grasping: The system enters the post-grasping phase when the grasp is executed and the tactile feedback at fingertips is available, the grasping system therein starts to benefit from having both visual information and tactile feedback. As it turns out in many applications, it is infeasible to capture dynamical properties of a system in precise mathematical forms when various uncertainties exist [14], whereas learning based methods perform very well if sufficient training data and a proper learning method are available [15], [16]. Therefore, we adopt a probabilistic model learned over both grasp parameters and corresponding tactile readings to represent the dynamics of the system. As shown in Fig. 2, our system estimates the grasp stability online through the feedback of tactile readings. When the executed grasp is observed as going to be unstable, our adaptation system is triggered to first try to adapt grasp forces at fingertips to keep the equilibrium of the grasp, and in cases when the disturbance is too large and force adaptation is insufficient, our system locally re-plans the location for one of the fingertips to keep the grasp stable. Note that the fingertip relocation planning is formulated as a single variable optimization problem, and it is computed by fast graph exploration and pruning in *HFTS*, it can be efficiently computed at 40 Hz online, which is crucial for closing the loop between the planning of fingertip relocation and control. This phase of the system is detailed in Sec. IV and Sec. V.

Compared with our previous work, we contribute several improvements, which were not feasible without this framework, to boost the robustness of this fingertip grasping system. Concretely, our system:

- Closes the loop of fingertip relocation planning and control to improve system robustness with respect to disturbance rejection.
- Adopts a Gaussian Process based cluster analysis for noise insensitive and compact *Hierarchical Fingertip Space* construction, which is additionally augmented in terms of object's local geometry and spatial relations to identify preferable contact locations.
- Optimizes grasp adaptability in terms of hand configurations to facilitate grasp adaptation system.
- Incorporates the augmented *HFTS* with grasp adaptation system to provide more accurate fingertip relocation as well as to avoid the risks introduced by blind decision making.

In a broad set of experiments, see Sec. VI, we show that the proposed system improves the robustness of fingertip grasping, and that it achieves the performance that can not be gained by either individual subsystems.

II. Related Work

Robotic grasping has seen considerable advances in the past decades, ranging from grasp synthesis and grasp evaluation

to sensorimotor control and adaptive execution, planning and control of robotic grasping have been addressed from various perspectives. In this section, we review related works in relation to the work of this paper.

A. Grasp Planning

Grasp synthesis has been studied for decades and still remains as a challenging one. Analytic methods and Data-Driven methods [6], [9] have been proposed to address this problem based on certain physical assumptions. Due to the differences in the nature of these two methodologies, they address this problem from very different angles.

Analytic methods commonly assume that sufficient and precise knowledge of the environment, such as object's geometry and its physical properties, is available to the system. Based on some low level geometrical features, e.g., contact positions and orientations, and by assuming some necessary physical properties of the workspace, e.g., rigid body, point contact, sufficient force and Coulomb friction etc., contact-level grasp synthesis is usually formulated as an optimization problem to generate stable grasps [2], [17]–[22], for which the grasp stability is commonly measured by force analysis in the contact wrench space [23]. Given contact locations on the object surface, efficient algorithms have been developed to calculate feasible hand configurations to realize desired contacts [3], [24]. For investigating how physical uncertainties affect the grasp stability, measurements such as grasp friction sensitivity [25] and independent contact regions [26] have been conceived to quantify the influence given by physical uncertainties. Furthermore, in order to plan grasps for certain motion requirements, contacts and hand configurations are planned with the concern of task compatibility and force conditions [27], [28]. As a basic function of in-hand manipulation, fingertip gaiting has been developed with rolling contact model and quasi-static assumption [29], [30]. As this type of methods usually look at low level geometric features of objects, it is often employed to generate fingertip grasps, for which precise contact locations play an important role.

Differently from the family of analytic methods, for enhancing the ability of understanding the objects and robotic hands from a more global view, data driven methods have been intensively researched to facilitate grasp planning. In order to efficiently find graspable parts of an object, 3D skeletal features have been recognized a useful tool. Representations such as Reeb Graph [31], Medial Axis [32], [33] and box decomposition [34] have been used to plan grasps with approach vector, hand opening and hand closing policies. Superellipsoids [35] and superquadrics [36]–[38], which are parametric models for shape approximation and decomposition, have been used for similar purpose to plan grasp parameters. Instead of geometrical models, topology analysis based loop grasping [39], [40] have been proposed to plan a specific type of caging using hand postural synergies. Furthermore, by the means of sufficient training data, probabilistic models have been developed to plan task oriented grasps [41]–[43]. Additionally, task oriented grasp stability can be assessed also in this framework when tactile information is available [1], [5].

For the grasp optimization in *HFTS* in this work, grasp synthesis is conducted in an analytic manner as contact-level grasping is considered. Nevertheless, it is also data driven since the construction of *HFTS* represents the shape of object in an multi-resolution way, so that both global and local features of the object are captured.

B. Grasp Control

The computation of proper grasping force has been investigated from many aspects, ranging from object geometry based analytic methods [27], [28] to learning based force optimization methods [44], [45]. For controlling both grasp positions and forces, hard switching methods have been devised to control one of them during different stages of the execution [46]. In the course of hybrid position and force control, grasp force and positions are controlled in different directions simultaneously [47]–[50] to explicitly model force and position between hands and objects. In order to allow robotic hands to interact with environments in a more robust way, impedance control, which regulates force implicitly by modeling a spring-mass-damper system, has been introduced into grasping to consider force, position, velocity and acceleration in a single framework [51]–[54]. Due to the lack of implicit position model, impedance control usually do not provide accurate positioning of fingertips, but instead performs better in regulating contact force during interactions with environments.

However, as the aforementioned works mainly focus on the control aspect of grasping, the merits of reusing the planner to support controller is not explored.

C. Disturbance Rejection

Disturbance rejection is essential in the control of dynamical systems [14], [55], which is especially the case for robots that work in unstructured and time variant environments. For robotic grasping systems, uncertainties and disturbances exist in many forms. Internally, inaccurate mathematical modeling of the physical processes, such as the oversimplified point contact model, makes both the grasp planning and control neglects some certain dynamical properties of the system. And externally, uncertainties such as external perturbations or the varying temperature of the end-effector, make the working space of the system even more nondeterministic. Therefore, we can not expect a grasping system to work properly without the ability of disturbance rejection.

A successful application of disturbance rejection in robotics is the balance control of legged robots [56]–[60], for which a common strategy for counteracting disturbances is to either adapt toques in actuators of different legs, or, in case of large disturbances for which torque adaptation is not enough to balance, to adapt leg configurations by local gaiting. Inspired by this line of research, we consider multi-fingered precision grasping falls in the same philosophy of legged robots in the context of disturbance rejection. For stabilizing a grasp under disturbances, we adopt the same strategy – when the grasp is estimated to be unstable, we stabilize the grasp by either adapting its contact forces or relocating the fingertips locally on the object surface.

III. Hierarchical Fingertip Space and Grasp Optimization

In this section, we elaborate on the *pre-grasping* phase of our system in the context of adaptive fingertip grasping. Based on the work of [13], we reformulate the *HFTS*, which was a search space defined for a single fingertip, to be a search space of multiple fingertips, so that the grasp optimization is presented in a more rigorous way. Furthermore, by exploring the geometric property and spatial relations of the fingertip space, an augmented fingertip space is developed to identify the quality of contact locations with respect to grasp adaptation. Next, we first describe the extraction and augmentation of fingertip space, which represents a contact-based grasping space that considers not only grasp stability, but also adaptability of contacts. Thereafter, we introduce a Gaussian Process based cluster analysis, with the aim of suppressing the effect of noise, for constructing the hierarchy of fingertip space. In the end of this section, we define a measure of grasp adaptability in terms of hand configurations and describe the grasp optimization algorithm in the framework of multilevel refinement in *HFTS*.

A. Extraction and Augmentation of Fingertip Space

Let us first define what is grasp planning in the proposed system. Denoted by $\mathcal{P} = \{p_i | p_i \in \mathbb{R}^3, i \in \{1, ..., n_p\}\}$ the observed object point cloud with n_p points, and given the number of fingertips, m, of the robot's hand. The goal of planning a fingertip grasp is to find m contact locations, $C_g = \{c_1, ..., c_m | c_i \in \mathcal{P}, i \in \{1, ..., m\}\}$, on the object surface represented by $\mathcal{P}$, so that C_g forms a grasp satisfying certain criteria and can be realized by a hand configuration $joint_g \in \mathbb{R}^d$, where d is the DoF of the robot's hand. Note that the number of potential contact locations is large, and many of them are not viable or preferred due to surface local geometry. In order to focus on viable contact locations during the grasp synthesis, we propose a notion called *Fingertip Space* to represent a finite set of locations on the object surface as contact candidates.

A *Fingertip Space* of $\mathcal{P}$ is denoted as $\Phi(\mathcal{P}) = \{\phi_1, ..., \phi_{n_f}\} \subset \mathcal{P}$, which is a set of admissible contact locations in $\mathcal{P}$ that satisfies certain criteria, and every element, ϕ_i, of this set is called a *Fingertip Unit*. Given different application scenarios, the criteria may vary correspondingly, e.g., if a caging grasp is desired, contacts at locally concave locations are preferred to provide unbreakable waypoints [61]. For the purpose of fingertip grasping while taking into account the physical properties of the fingertip, which is a SynTouch sensor in this work, we consider locations that have large local curvatures not viable for a stable fingertip contact. In other words, as shown in Fig. 4, the locations that are at edges or sharp shape regions clearly can not safely stabilize fingertip contacts.

Therefore, we establish a local surface curvature based filter to extract $\Phi(\mathcal{P})$ from $\mathcal{P}$. Let $N^r(p_i) \subset \mathcal{P}$ denote the set of points around p_i within one fingertip size[2] r, and denoted by $\mathcal{K}(N^r(p_i))$ the local surface curvature estimated from $N^r(p_i)$, the fingertip space of object $\mathcal{P}$ is extracted as shown in Eq. 1:

$$\Phi(\mathcal{P}) = \{\phi_i \mid \mathcal{K}(N^r(\phi_i)) \leq \lambda, \phi_i \in \mathcal{P}\} \tag{1}$$

where $\lambda \in \mathbb{R}$ is a curvature threshold determined empirically in terms of the physical properties of the fingertip. In the rest of this paper, we refer to Φ as short for $\Phi(\mathcal{P})$ where it does not cause confusion.

As discussed in Sec. II-C, for the purpose of disturbance rejection during grasp execution, our system adapts the locations of fingertips to keep the grasp stable when grasp force adaptation is not enough. The fingertip space therefore needs to take into consideration whether a location on the object affords enough space around it to allow fingertip relocations, i.e. if a fingertip is close to large curvature locations, adapting this fingertip into those directions has a risk of completely losing contact for this fingertip. For this, rather than removing those locations from the fingertip space, we still accept those locations as fingertip units with, however, an attached penalty term as a logistic function, which penalizes risky locations and treats the rest locations equally. Let $c(\phi_i) \in \mathcal{P} \setminus \Phi$ be the closest point to fingertip unit ϕ_i that has been rejected by Eq. 1, the penalty $w_i \in \mathbb{R}$ attached to fingertip unit ϕ_i is computed as:

$$w_i = \frac{1}{1 + e^{-\gamma \left\| \phi_i - c(\phi_i) \right\|}} \tag{2}$$

where $\gamma \in \mathbb{R}^+$ is an elasticity factor that controls how much we want to keep the fingertips away from the risky regions. An example fingertip space extraction of a spray is shown in Fig. 4, red points mark the locations rejected by fingertip space and colors on the fingertip units show the penalties attached to the units.

B. Problem Approximation of Fingertip Space

The *Fingertip Space* defines a solution space of contact locations for fingertip grasping. However, given some common objective functions of grasp optimization, such as grasp quality measure, this space renders highly non-convex shape that contains many local optimum [2]. Moreover, consider also the number of possible solutions and maybe even more complicated objective functions, it is impractical to apply optimization routines directly on this problem. A feasible strategy for handling this kind of problems is to apply Surrogate Models or multilevel refinement [62], [63], which recursively approximates the original problem into a hierarchy of simpler tractable problems, i.e. surrogate models, and then optimizes and updates the solution in a reversed way until a solution is obtained for the original problem.

For applying this strategy to the grasp optimization problem, we therefore first need to define how to approximate our problem into its surrogates. As in fingertip grasping, the grasp quality in the wrench space is determined by the local geometry of contacts, i.e. locations and normal orientations, and it is proven that the grasp quality is bounded by a Lipschitz constant given bounded variations of local geometry of contacts [64]. We hence, in terms of the similarity of contact local geometry, propose a method for constructing the hierarchy of

[2]For the SynTouch sensor used in this work, the fingertip size r is $14mm$, http://www.syntouchllc.com/

approximated surrogates for the grasp optimization problem. Next, we will first explain the definition of the hierarchy of surrogates in the context of fingertip grasping, and then elaborate on the details of the construction of it.

Given a *Fingertip Space* Φ, which is the original solution space of contact locations for fingertip grasping, we construct a hierarchy of it as a similarity based graph $G_\Phi = (E_\Phi, V_\Phi)$ with the induced hierarchy levels $i \in \{1, ...l\}$. Concretely, Φ is recursively partitioned into smaller sets of fingertip units, denoted as $\hat{\phi_{i,j}} \subset \Phi$, and is represented as a node $\phi_{i,j} \in V_\Phi$ in graph G_Φ, where i represents the level of $\phi_{i,j}$ in the hierarchy and j represents the index of the partition in level i. As will be considered by grasp optimization later, we herein define the location and orientation of a set of fingertip units as $\mathbf{p}(\hat{\phi_{i,j}}) \in \mathbb{R}^3$ and $\mathbf{n}(\hat{\phi_{i,j}}) \in \mathbb{R}^3$, which are the mean location and orientation of all the fingertip units contained in $\hat{\phi_{i,j}}$, and they are used to quantify the geometry property represented by this sets of fingertip units. Additionally, in terms of the penalties assigned to fingertip units during the fingertip space augmentation, the penalty assigned to a node $\phi_{i,j}$ is defined as shown in Eq. 3.

$$w_{i,j} = \frac{1}{|\hat{\phi_{i,j}}|} \sum_{\forall \phi_k \in \hat{\phi_{i,j}}} w_k \tag{3}$$

The recursive partitioning is processed in a top-down manner until the bottom level of G_Φ, where the fingertip units reside, is reached and a partitioning tree is constructed. And then based on this partitioning tree, which already have edges defined between nodes and their parent nodes, we establish extra connectivities E_Φ for Φ in terms of the number of hops, denoted as $hop(\phi_{i,j}, \phi_{i,k})$, between nodes $\phi_{i,j}$ and $\phi_{i,k}$ in the same level i as in Eq. 4.

$$E_\Phi = \{\{\phi_{i,j}, \phi_{i,k}\} | hop(\phi_{i,j}, \phi_{i,k}) \leq h, \phi_{i,j}, \phi_{i,k} \in V_\Phi\} \tag{4}$$

where h is the distance limit for nodes to be neighbors, we set $h = 4$ in this work to construct the graph. As shown in Fig. 3, the higher level a node is located, the longer Euclidean distance it travels to reach its neighbors.

As shown in Fig. 3, we can now define the i-th surrogate approximation of G_Φ as:

$$\begin{aligned} (G_\Phi)_i &= ((V_\Phi)_i, (E_\Phi)_i) \\ (V_\Phi)_i &= \bigcup_j \phi_{i,j} \\ (E_\Phi)_i &= \{\{\phi_{i,j}, \phi_{i,k}\} \mid \{\phi_{i,j}, \phi_{i,k}\} \in E_\Phi\} \end{aligned} \tag{5}$$

which is a subgraph of G_Φ and represents an approximated problem of contact locations in lower resolution. As we can see in Fig. 3, the larger i is, the lower resolution we have for the problem.

For the recursive partitioning of Φ, in order to enable handling of noise inherit to the sensory data, we find partitioning centers by employing a *Gaussian process* based filter, derived from [65], to recursively repartition Φ into smaller sets of fingertip units. As both the location and the orientation of fingertip units are important for grasp planning, in our GP based filter, a point is selected as a partition center if it has a large location uncertainty with respect to the estimated object surface, or the curvature near the point varies importantly.

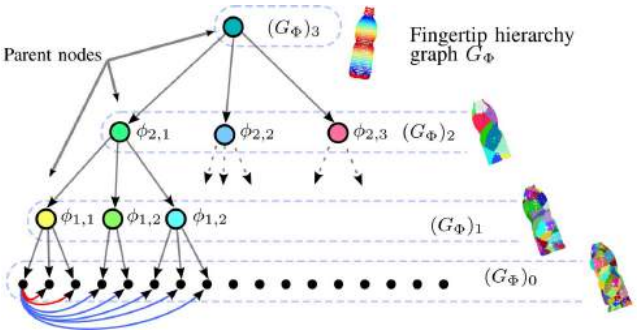


Fig. 3. Surrogate approximations represented as a graph. Fingertip unit partitions are represented as nodes in different levels of the hierarchy and the connectivities in this graph are represented by edges.

Note that the partitioning is done recursively on each level and on each partition, one GP model is required every time when partitioning is needed. Using the GP filter from [65], when a repartitioning of $\hat{\phi_{i,j}}$ is required to produce a lower level in the hierarchy, iterative input point for the GP filter is a fingertip unit $\mathbf{x} \in \hat{\phi_{i,j}}$, and the output of it includes the normal direction $\mathbf{n} \in \mathbb{R}^3$ associated with $\mathbf{x}$, and the signed distance $y \in \{-1, 0, 1\}$ which represents whether a point is inside, on or outside the estimated object surface. The GP filter is then used to predict the output mean value and its associated variance as in E.q 6 [66], in our case, we take the variance(uncertainty) of the signed distance $v(\mathbf{y})$ and the mean value of the normal direction $m(\mathbf{n})$ to select the partition centers for the next level.

$$[m(\mathbf{y}), v(\mathbf{y}), m(\mathbf{n}), v(\mathbf{n})] = GP(\mathbf{x}) \tag{6}$$

In order to capture the shape variance of object surface, we adopt the *thin plate spline* [67] as the kernel function for GP, as shown in E.q 7:

$$k(x_i, x_j) = 2\left\|x_i - x_j\right\|^3 - 3\Omega\left\|x_i - x_j\right\|^2 + \Omega^3 \tag{7}$$

where $\Omega \in \mathbb{R}$ is computed by:

$$\Omega = \underset{i,j}{\operatorname{argmax}} \left\|x_i - x_j\right\|, \quad x_i, x_j \in GP \tag{8}$$

A point $\mathbf{x}_* \in \hat{\phi_{i,j}}$ is selected as a partition center for next level if $v(\mathbf{y}_*) > V_{thresh}$ or $m(\mathbf{n}_*)^T \mathbf{n}_* < \theta_{thresh}$. V_{thresh} and θ_{thresh} are two parameters selected online during partitioning, and are used to control the number of output centers. For deciding the number of partitioning centers for each level, we apply the exponential function from [2], with the second top level having 20 base partitions and the rest levels having 4 partitions from each of their parent level. As the number of fingertip units varies between different partitions, the repartitioning of a branch in the partitioning tree is terminated when $|\hat{\phi_{i,j}}| \leq 10$, and all the fingertip units contained in that partition is directly extended to the next level without partitioning until all branches terminate. In this way, we can guarantee a balanced partitioning tree, and hence a valid surrogate approximation for every level in the hierarchy.

An example of GP clustering on the original fingertip space of a spray model is shown in Fig. 4, the fingertip units are partitioned with respect to the selected centers by $\mathcal{L}^2$ distance in $\mathbb{R}^3$. We can see that the partition centers are more dense in areas where the surface changes more rapidly, so as to capture its shape variance. Therefore, the shape local geometry is captured, as needed by grasp synthesis, so that fingertip units are partitioned in terms of the similarities of locations and orientations.

Fig. 4. Left: fingertip extraction with attached penalties. Middle: GP for the spray model represented by 20 fingertip units. Right: partitioning result.

C. Hierarchical Fingertip Space

Having constructed the hierarchy of surrogate approximations G_Φ, which is a hierarchical solution space for locations of fingertip contacts, we can now proceed to investigate the synthesis of fingertip grasps. As synthesizing a fingertip grasp involves multiple fingertip contacts, it is indeed a combinatorial optimization problem. Therefore, based on the solution space of contacts G_Φ, let us define the hierarchy of surrogate approximations, which is the hierarchical solution space $\overline{G_\Phi} = (\overline{V_\Phi}, \overline{E_\Phi})$ for a fingertip grasp, call a *Hierarchical Fingertip Space*, with m contacts as in Eq. 9:

$$\begin{aligned} \overline{G_\Phi} &= G_\Phi^1 \times ... \times G_\Phi^m \\ \overline{V_\Phi} &= \bigcup_{i,j} \overline{\phi_{i,j}} \\ \overline{\phi_{i,j}} &= (\phi^1_{i,j_1}, ..., \phi^m_{i,j_m}) \end{aligned} \tag{9}$$

where $G_\Phi^k = (V_\Phi^k, E_\Phi^k)$ is the surrogate hierarchy for the k-th fingertip, and the nodes $\overline{\phi_{i,j}}$ are tuples composed from nodes of $\phi^k_{i,j_k} \in V_\Phi^k$ from each G_Φ^k in level i and it represents m contact locations defined in fingertip space. According to the penalties assigned to nodes of G_Φ^k in Eq. 3, we define the penalty for a tuple of contacts as:

$$\overline{w_{i,j}} = \mathbf{max}\{w^1_{i,j_1}, ..., w^m_{i,j_m}\} \tag{10}$$

which takes the maximum penalty from all contacts to avoid the risk of having any contacts to be located at risky locations. Observe that the definition in Eq. 9 implies that different fingertips can have different fingertip spaces. In this work, as we have the same SynTouch sensors at all fingertips, the fingertip spaces are the same for all fingertips.

From the definition in Eq. 9 we can see that $\overline{G_\Phi}$ is a graph with vertices, however, representing combinations of contacts rather than single contacts. In order to search in $\overline{G_\Phi}$, we also need to define the connectivity $\overline{E_\Phi}$ in the graph. As the *HFTS* $\overline{E_\Phi}$ inherits the hierarchy levels $i \in \{1, ..., l\}$ from the G_Φ, the connectivities in this graph composes two types of edges: 1) edges between nodes and their parent nodes, and 2) edges connecting nodes in the same level i. Benefiting from the well defined G_Φ, these two types of connectivities can be defined in a unified form as:

$$\begin{aligned} \overline{E_\Phi} = \{(\overline{\phi_{i_1,j_1}}, \overline{\phi_{i_2,j_2}}) \mid \forall k \in \{1, ..., m\}, \\ \{\overline{\phi_{i_1,j_1}}^{(k)}, \overline{\phi_{i_2,j_2}}^{(k)}\} \in E_\Phi\} \end{aligned} \tag{11}$$

where $\overline{\phi_{i_1,j_1}}^{(k)} \in V_\Phi^k$ is the k-th item of tuple $\overline{\phi_{i_1,j_1}}$. From this definition, we can know that two grasp solutions $\overline{\phi_{i_1,j_1}}$ and $\overline{\phi_{i_2,j_2}}$ are connected if all the items of them have connected parent nodes in each G_Φ^k, or if all the items have connected nodes in the same hierarchy level i in each G_Φ^k. Therefore, it is clear that $|i_1 - i_2| \leq 1$.

Similarly to the surrogate models in Sec. III-B, we can now define the i-th surrogate approximation of fingertip grasping in *HFTS* as in Eq. 12. Observe that in this approximation, each grasp solution $\overline{\phi_{i,j}}$ is an approximated solution to the original fingertip grasping problem. Having defined the *HFTS*, we will next proceed to explain how the grasp optimization is conducted in it using surrogate-based approximation.

$$\begin{aligned} (\overline{G_\Phi})_i &= ((\overline{V_\Phi})_i, (\overline{E_\Phi})_i) \\ (\overline{V_\Phi})_i &= \bigcup_j \overline{\phi_{i,j}} \\ (\overline{E_\Phi})_i &= \{\{\overline{\phi_{i,j}}, \overline{\phi_{i,k}}\} \mid \{\overline{\phi_{i,j}}, \overline{\phi_{i,k}}\} \in \overline{E_\Phi}\} \end{aligned} \tag{12}$$

D. Grasp Optimization in HFTS

As discussed in Sec. III-B, when an optimization problem is too complicated or expensive, it is intractable to apply optimization routines directly on it, such as the synthesis of fingertip grasp we encounter in this work. To address such a problem, we propose to apply surrogate-based optimization, which is also called multilevel refinement [62], [63], in the *HFTS* to make this problem tractable.

In the *HFTS*, a grasp solution is defined as a vertex $\overline{\phi_{i,j}}$, which is a combination of contacts on the object surface. However, in order to optimize grasps while considering potential hand configurations for realizing selected contacts, we herein define a grasp solution to be a tuple $g^i = (\overline{\phi_{i,j}}, joint_g)$, $joint_g \in \mathbb{R}^d$ in the i-th surrogate model $(\overline{G_\Phi})_i$ for the grasp optimization below. Next, before we discuss about how to optimize a grasp, let us first define what objective function is employed in the grasp synthesis.

1) Grasp Stability: As the grasp optimization procedure is in the pre-grasping phase where we have only visual information of object represented by $\overline{G_\Phi}$, it is therefore convenient to evaluate grasp stability using contact based force closure analysis [17], [23]. Given a grasp g^i with contact locations $\overline{\phi_{i,j}}$, the grasp quality measure $Q(\overline{\phi_{i,j}}) \in \mathbb{R}$ is evaluated as the minimum offset between the origin of the wrench space and the convex hull spanned by friction cones of contacts of $\mathbf{p}(\hat{\phi_{i,j}})$ and $\mathbf{n}(\hat{\phi_{i,j}})$ [18].

2) Grasp Reachability: Due to the kinematic constraints of robotic hand, the optimization is also constrained by the hand kinematics, i.e. the selected contacts $\overline{\phi_{i,j}}$ of grasp g^i are either reachable: $R^*(\overline{\phi_{i,j}}) = 0$, or unreachable: $R^*(\overline{\phi_{i,j}}) = 1$, where $R^*(\overline{\phi_{i,j}})$ is a binary reachability constraint. Since a robotic hand can have many degree of freedoms, it is not feasible to compute $R^*(\overline{\phi_{i,j}})$ during the optimization. However, this constraint can be linearly relaxed to be a measure of how dissimilar a grasp g^i is compared to the closest viable hand configuration $joint_g^*$ of g^* that achieves $\overline{\phi_{i,j}}^*$, and then the reachability measure of a m fingered grasp g^i is reformulated as a reachability residual by $R(\overline{\phi_{i,j}}) \in \mathbb{R}^+$ to be minimized:

$$R(\overline{\phi_{i,j}}) = \left\| m_g - m_{g^*} \right\| \tag{13}$$

where $m_g \in \mathbb{R}^{6\times(m-2)}$ is an affine invariant encoding of $\overline{\phi_{i,j}}$ of grasp g^i in terms of its contact locations and normals [13]. By randomly sampling the hand configurations and saving the corresponding m_g and hand configuration $joint_{g^i}$ into a structured lookup table T in offline, for which we adopt a k-d tree like data structure for $O(logn)$ query time, we can rapidly compute the reachability residual $R(g^i)$ with contacts $\overline{\phi_{i,j}}$ by querying the lookup table online during grasp optimization and at the same time get the hand configuration for realizing the contacts, the query response of table T is shown in Eq. 14.

$$T \ : \ \overline{\phi_{i,j}} \rightarrow (joint_g^*, R(\overline{\phi_{i,j}})) \tag{14}$$

3) Grasp Adaptability: Recall that in order to approach disturbance rejection, as discussed in Sec. II-C, our system adapts fingertip locations to counteract large disturbances. This requires a hand to be able to move fingertips tangentially on the object surface. By decomposing the hand Jacobian and calculating the manipulability [68] of a hand configuration in the tangential plane of contacts, we measure the *adaptability*, denoted as $A(joint_g) \in \mathbb{R}^+$, for a grasp g^i with hand configuration $joint_g$ by measuring its ability of moving fingertips tangentially along the object's surface. Concretely, given the Jacobian $J_f(joint_g) \in \mathbb{R}^{3\times n}$ and normal $\mathbf{n}_f$ of fingertip f, the Jacobian can be rotated by $R_f \in \mathbb{R}^{3\times3}$ such that the last row of $\hat{J}_f(joint_g) = R_f J_f(joint_g)$ corresponds to the movement of fingertip in the direction of $\mathbf{n}_i$. The first two rows of $\hat{J}_f$, denoted by $\widetilde{J_f}(joint_g) \in \mathbb{R}^{2\times n}$, is the projection of J_f in the tangential plane of contact. Assuming that the hand base won't move during adaptation, we define *adaptability* of a grasp g^i as shown in Eq. 15. An example of grasp adaptability measure is shown in Fig. 5. It is important to note that since this measure is hand configuration based, it is affine invariant, and hence grasp pose independent.

$$A(joint_g) = \sum_f \sqrt{det\widetilde{J_f}(joint_g)\widetilde{J_f}^T(joint_g)}$$

In order to capture all the above three objectives in the grasp optimization, the objective function $\theta(g)$ of grasp optimization is defined as:

$$\theta(g) = Q(\overline{\phi_{i,j}}) - \alpha R(\overline{\phi_{i,j}}), \quad \alpha \in \mathbb{R}^+ \tag{15}$$

where α is a weighing factor to account for the hand size.

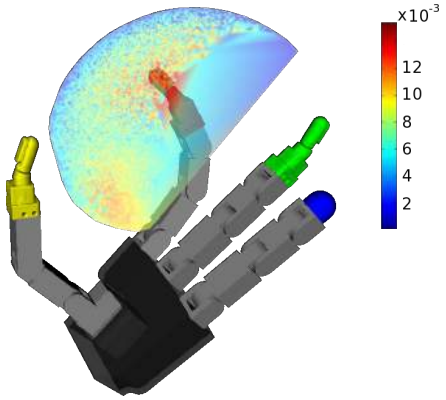


Fig. 5. Grasp Adaptability example for fingertip 1.

Notice that the adaptability $A((joint_g))$ is not directly employed in the object function $\theta(g)$. In practice, we adopt a prioritized lookup table for $R(g)$, which has been successfully used in inverse kinematics computation and computer graphics applications [69], [70], to use adaptability $A(\overline{\phi_{i,j}})$ as a reference to prioritize the lookup process of $R(g)$ when querying reachability residual. As we can see in Fig. 6, for the same contact locations, there can be multiple hand configurations for realizing it, however, our prioritized lookup table will always return the hand configuration with the best adaptability.

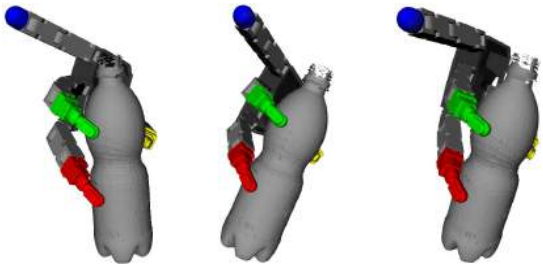


Fig. 6. Grasps with same contacts and different adaptabilities. The left most grasp is the most adaptable grasp given these 3 contacts.

Having defined the optimization objective function $\theta(g)$, we can now proceed to synthesize grasps by grasp optimization. For conducting grasp optimization using the surrogate-based optimization metaheuristic, which requires an optimization routine for finding solutions of the surrogate approximations, the stochastic hill climbing is adopted to escape from local optimum by means of randomness. In this algorithm, the objective function is not directly considered to improve solutions, instead, the switch between two solutions g^i and g^i_{new} is determined by the probabilistic function in Eq. 16.

$$Pr(g^i, g^i_{new}) = \left(1 + \exp \frac{\frac{1}{w_{g^i}}\theta(g^i) - \frac{1}{w_{g^i_{new}}}\theta(g^i_{new})}{\zeta}\right)^{-1} \tag{16}$$

where w_{g^i} is the penalty assigned to a tuple of contacts defined by Eq. 10. Note that the search randomness is determined by ζ, which makes the optimization more random when a larger value is chosen. The grasp optimization algorithm is shown in Alg. 1, note that this algorithm produces both contact

positions C_g and a hand configuration $joint_g$ to realize them. Since our reachability encoding is affine invariant, the hand pose is simply calculated by affine transform between contact locations and fingertip positions of selected hand configuration $joint_g$, in case of the reachability residual $R(\overline{\phi_{i,j}}) \neq 0$, a local optimization of joint configuration by linear interpolation [71] is adopted to realize precise contacts.

Algorithm 1 Surrogate-Based Optimization in *HFTS*

Input: $stopCondition(g^i)$, $\overline{G_\Phi}$, $maxIter$
Output: grasp $g = (C_g, joint_g)$

for $i = l-1$ to 0 **do**
  if $i == l-1$ **then** ▷ Initialization
    $\overline{\phi_{i,j}} \leftarrow$ random from $(\overline{G_\Phi})_i$
    $(joint_g^*, R(\overline{\phi_{i,j}})) \leftarrow T(\overline{\phi_{i,j}})$
    $g^i \leftarrow (\overline{\phi_{i,j}}, joint_g^*)$
    $g \leftarrow g^i$
  else ▷ Extend to Lower Surrogate
    $\overline{\Phi_{i+i,j}} \leftarrow$ contacts of g^{i+1}
    $\overline{\phi_{i,j}} \leftarrow$ random child of $\overline{\phi_{i+1,j}}$
    $(joint_g^*, R(\overline{\phi_{i,j}})) \leftarrow T(\overline{\phi_{i,j}})$
    $g^i \leftarrow (\overline{\phi_{i,j}}, joint_g^*)$
    $g \leftarrow g^i$
  end if
  for 1 to maxIter **do** ▷ Optimize Surrogate
    $\overline{\Phi_{i,j}} \leftarrow$ contacts of g^i
    $\overline{\phi_{i,j}} \leftarrow$ random neighbor of $\overline{\phi_{i,j}} \in (\overline{G_\Phi})_i$
    $(joint_g^*, R(\overline{\phi_{i,j}})) \leftarrow T(\overline{\phi_{i,j}})$
    $g^i \leftarrow (\overline{\phi_{i,j}}, joint_g^*)$
    if $Pr(g, g^i) \geq \text{rand}(0,1)$ **then**
      $g \leftarrow g^i$
    end if
    $g^i \leftarrow g$
    if $stopCondition(g^i)$ **then** ▷ Good Solution
      break
    end if
  end for
end for

In Alg. 1, random neighbors and random children of a node $\overline{\phi_{i+1,j}} \in \overline{G_\Phi}$ are found by the edges $\overline{E_\Phi}$ of graph $\overline{G_\Phi}$ and the surrogate approximation in Eq. 12. During grasp optimization, when the agent is satisfied with the current solution on a surrogate approximation, one can simply stop optimizing on the current level and start to optimize in the next level where the problem has better resolution. The $stopCondition(g^i)$ function in Alg. 1, which outputs boolean values to decide whether to stop on the current surrogate, and can be customized dependent on user's requirements. If it outputs $false$ only, the optimization will carry on until the $maxIter$ reaches. Some examples of $stopCondition(g^i)$ is shown in Fig.7. We can see that when we have a loose stop condition for a surrogate, optimized grasps can have many different shapes and qualities. However, as we observed in many experiments, if we keep optimizing until $maxIter$ reaches for every surrogate, the grasps will always have very similar shapes, as shown in the left most of Fig. 7, the grasp will try to span fingertips over a large area on the object with contacts at low penalty locations, and the hand will be posed perpendicular to the main axis of the object.

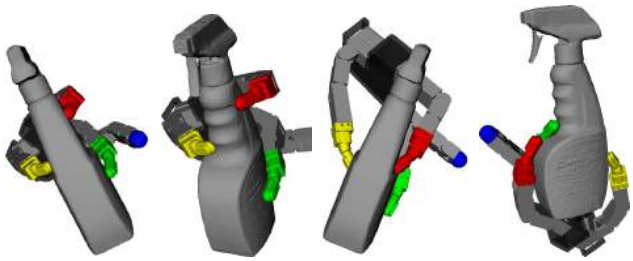


Fig. 7. Example grasps on spray model. Left: $stopCondition(g^i)$ outputs always $false$. The rest grasps are optimized with a stop condition that, as soon as the the grasp is stable and reachability residual smaller than 0.006, the agent stops optimizing on the current surrogate.

IV. Modeling Grasping Dynamics

Once a grasp is synthesized, our system will execute the grasp and enter the post-grasping phase, and importantly, this is when tactile feedback kicks in. As discussed in Sec. I, in many dynamical systems, it is infeasible to represent all the complicated physical processes in precise mathematical forms, especially when various uncertainties exist [14]. However, as has been adopted by many robotic systems [15], [16], learning based methods are capable of capturing useful properties of such systems if sufficient training data is provided. In this section, we introduce our in-hand grasp adaptation strategy based on an in-hand virtual frame representation of fingertip grasping, as well as explaining the corresponding probabilistic model, which is used to represent dynamical properties of the system, for online grasp adaptation.

A. Grasp Representation in Virtual Frame

During the grasp execution, the equilibrium of a grasp is mainly determined by the relative locations and forces exerted by fingertips. Therefore, after the fingertips are positioned at the synthesized contacts, we hand over dynamic control of the system to a robust impedance controller [4] to regulate grasp stiffness implicitly, in both contact normal and tangential directions, and for this, a contacts based virtual frame (VF) is needed. In this work, we equip the first 3 fingertips of the Allegro hand with SynTouch sensors and evaluate our system by this 3-fingered grasping system. As shown in Fig. 8, the VF is constructed by the location of 3 fingertips as defined in Eq. 17. As the hand configuration can change when fingertip relocation is applied for disturbance rejection, this virtual frame is updated online during the grasp execution.

$$
\begin{aligned}
R_o &= [\mathbf{v}_x, \mathbf{v}_y, \mathbf{v}_z] \in \text{SO(3)} \\
\mathbf{v}_x &= \frac{\mathbf{p}_3 - \mathbf{p}_1}{\|\mathbf{p}_3 - \mathbf{p}_1\|} \\
\mathbf{v}_z &= \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times \mathbf{v}_x}{\|(\mathbf{p}_2 - \mathbf{p}_1) \times \mathbf{v}_x\|} \\
\mathbf{v}_y &= \mathbf{v}_z \times \mathbf{v}_x
\end{aligned}
\tag{17}
$$

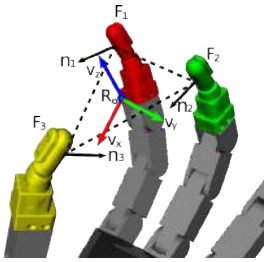


Fig. 8. Virtual frame defined by fingertip contacts.

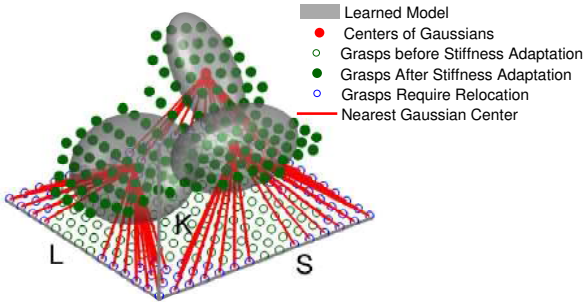


Fig. 9. GMM model Θ for grasp stability estimation and decision making for grasp adaptation.

where $\mathbf{p}_1, \mathbf{p}_2$ and $\mathbf{p}_3 \in \mathbb{R}^3$ denotes the locations of there fingertips respectively.

Differently from the pre-grasping phase in which a grasp is represented by contacts on the object and the corresponding hand configuration, in the controller of this system, in order to model the dynamic behavior of the hand together with the tactile feedback, a grasp is represented in the VF as $\hat{g} = (K, L, S)$ with its stiffness $K = (K_1, K_2, K_3) \in \mathbb{R}^3$, rest length $L = (L_1, L_2, L_3) \in \mathbb{R}^3$, which is defined as the distance between each fingertip and the center of VF, and tactile readings $S = (S_1, S_2, S_3) \in \mathbb{R}^{57}$.

B. Grasp Adaptation

Various disturbances exist in the grasp execution, such as visual perception errors, fingertip positioning errors and external perturbations etc. Therefore, the system should be aware of when the grasp is not stable and then adapt the grasp accordingly to keep the stability.

Extending from our previous work [5], a VF based stability estimator is employed to classify stable and unstable grasps in terms of K, L and S. By training a Gaussian Mixture Model Θ over a variety of grasps represented in the virtual frame on different objects with different weights, see Sec. VI, the distribution of stable grasps is modeled as a probability function shown in Eq. 18 and the estimator predicts the grasp stability by checking the likelihood $p(\hat{g}|\Theta)$.

$$p(\hat{g}|\Theta) = \sum_{i=1}^{n_g} \pi_i \mathcal{N}(\hat{g}|\mu_i, \Sigma_i) \tag{18}$$

where n_g is the number of Gaussian components, each of which has a prior π_i. $\mathcal{N}(\hat{g}|\mu_i, \Sigma_i)$ is the Gaussian distribution with mean μ_i and covariance Σ_i.

When a grasp is predicted as unstable in terms of the tactile feedback, our adaptation system, as shown in the post-grasping part in Fig. 2, will try to keep the equilibrium of the grasp by either grasp force adaptation or fingertip local relocation.

The same as our previous work [5], by computing the the minimum Mahalanobis distance from the grasp $\hat{g}$ to the center of each Gaussian component, we consider the grasp $\hat{g}$ belonging to the learned model Θ if $\hat{g}$ is inside two standard deviations of any Gaussians in Θ. In this case, the grasp force adaptation is triggered to indirectly regulate contact forces by manipulating grasp stiffness K, which is obtained by computing maximum expectation of K conditioned on grasp rest length L and tactile readings S.

However, when the minimum Mahalanobis distance implies that our system has not experienced this grasp $\hat{g}$ during the training, which means that the tactile feedback and the corresponding grasp rest length is very different from the learned model Θ, the adaptation system will then try to relocate one of its fingertips on the object surface to change the grasp rest length, such that the unstable grasp can be dragged back to a stable stage. As will be detailed in Sec. V, the fingertip to be relocated and the new location of it is computed in the *HFTS*. An example of the model Θ and how it is used to determine different stages of the system is shown in Fig. 9.

V. Fingertip Relocation by Local Exploration

When the grasp $\hat{g}$ encounters some large disturbances, the tactile readings S for the grasp $\hat{g} = (K, L, S)$ would change rapidly and hence the grasp $\hat{g}$ will be shifted out from the distribution Θ. In this case, grasp stiffness adaptation is not enough to balance the grasp. However, as discussed in Sec. II-C it is intuitive that if one of the fingertips can be relocated to counteract the perturbations by applying force at a different location, there is still chances to keep the stability. Next, we explain an algorithm for determining which fingertip to relocate and where to relocate it using both visual representation *HFTS* and tactile sensing. And then we describe how we in practice implement the fingertip relocation.

A. Fingertip Relocation *in* HFTS

Recall that in the pre-grasping phase, grasp synthesis is formulated as a combinatorial optimization problem in the *HFTS*. As fingertip relocation is a problem of finding reachable contacts on the object that can counteract instability, we therefore formulate the fingertip relocation also as an optimization problem in the *HFTS*. However, differently from the grasp synthesis, the purpose here is to drag the grasp $\hat{g}$ with contacts $\overline{\phi_{i,j}}$ from its current unstable stage to a new grasp that is within the distribution Θ. For this, we find the closest center of Gaussian components, denoted as $\hat{g}^* = (K^*, L^*, S^*)$, of $\hat{g}$ in Θ and use the rest length of it to define our objective function to be minimized as shown in Eq. 19.

$$\theta^*(\overline{\phi_{i,j}}) = \left\| \hat{L} - \hat{L}^* \right\| + \beta R(\overline{\phi_{i,j}}) \tag{19}$$

where $R(\overline{\phi_{i,j}})$ is still the reachability defined in Eq. 13 and $\beta \in \mathbb{R}^+$ is a weighing factor to account for the hand size.

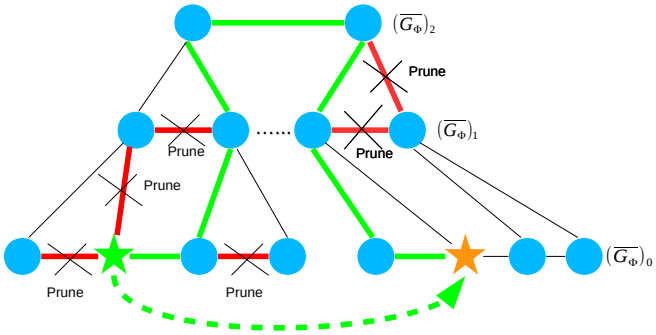


Fig. 10. Flood fill in *HFTS* for fingertip relocation optimization. The green path shows how the search fringe evolves, and the red edges show the pruned path due to the 2 criteria defined in Alg. 2.

The minimization of Eq. 19 now boils down to a problem of single variable optimization in the graph $\overline{G_\Phi}$, and the variable to be optimized is the new location of one of the fingertips. In this work, the fingertip to be relocated is chosen between $F1$ and $F2$, as shown in Fig. 8, since relocating the thumb $F3$ will cause potentially more instability. Our strategy of choosing the adapted fingertip is straight forward: the system computes the optimization for $F1$ and $F2$ in parallel and the one with better final optimized value of Eq. 19 wins. In this work, the optimization for the f_o-th fingertip is done by flood fill, which is a breath-first metaheuristic, in graph $\overline{G_\Phi}$.

As fingertip relocation is done locally in the *HFTS* for only one fingertip, there are some properties can benefit our optimization procedure. Firstly, when the reachability measure increases too much in a search fringe, measured by a tolerance factor ϵ_R, this fringe can be pruned as the outer space is more unreachable. Secondly, recall that a node $\overline{\phi_{i,j}}$ in graph $\overline{G_\Phi}$ represents m contact locations, this means that if we switch our current solution from one node to another during optimization, all m contact locations can change. However, as only one fingertip is going to be relocated, the rest fingertip should be kept at the same location. Therefore, comparing to Alg. 1, there is an extra rule the optimization routine must respect, denoted by ϕ_{old} the node that represents the current grasp contacts, and denoted by ϕ_{new} a new solution that may provide better objective value, this rule is defined as in Eq. 20.

$$Prune(\phi_{old}, \phi_{new}, f_o) = \begin{cases} True, & \exists i \neq f_o, \phi_{old}^{(i)} \not\subseteq \phi_{new} \\ False, & \text{otherwise.} \end{cases} \tag{20}$$

where f_o is the index of the fingertip to be relocated. Note that as the search fringe can go upwards in the hierarchy graph $\overline{G_\Phi}$, this rule asserts that the fingertip contacts that should be kept fixed always exist along the optimization path. The optimization procedure is summarized in Alg. 2, we can see that the search is still augmented by the penalty factor $w_{\phi_{new}}$ defined in Eq. 10 to avoid not risky locations. An example of this algorithm is depicted in Fig. 10, due to the definition of *HFTS*, the search will always end up at the bottom layer where finer solutions reside, which are at least as good as the best solutions from upper layers. It is interesting to notice that, as this optimization routine can go upwards in the hierarchy, it also benefits from the surrogate approximations.

Algorithm 2 Fingertip Relocation by Optimization in $\overline{G_\Phi}$

Input: $\overline{G_\Phi}$, ϕ_{old}, ϵ_R, f_o
Output: $\hat{p}$ ▷ New Location

$R_o = R(\phi_{old})$
$\phi^* \leftarrow \phi_{old}$
$Queue.push(\text{neighbors of } \phi_{old})$
while Queue is not empty **do**
  $\phi_{new} \leftarrow Queue.pop()$
  if $R(\phi_{new}) > R_o + \epsilon_R$ **then**
    continue ▷ Pruning
  else if $Prune(\phi_{old}, \phi_{new}, f_o)$ **then**
    continue ▷ Pruning
  end if
  if $\frac{1}{w_{\phi_{new}}}\theta^*(\phi_{new}) < \frac{1}{w_{\phi^*}}\theta^*(\phi^*)$ **then**
    $\phi^* \leftarrow \phi_{new}$
    $Queue.push(\text{neighbors of } \phi_{new})$ ▷ Flood Fill
  end if
end while
  $\hat{p} \leftarrow \phi^{*(f_o)}$
return $\hat{p}$

B. Fingertip Relocation in Practice

Once the f_o-th fingertip is determined by Alg. 2 to be relocated to location $\hat{p}$, the adaptation system will then start to move it towards $\hat{p}$. However, there are two problems in practice: 1) Perturbations usually occur in a period of time with different directions and strengths and the grasp status accordingly changes rapidly. A new grasp adaptation might be required before the current fingertip relocation is finished. And 2) In order to not switch between impedance control and position control frequently, the system will stay in impedance control mode during relocation. Nonetheless, we also want the movement of the fingertip to be compliant to the object surface to be able to measure the stability, so as the impedance controller will always try to keep the equilibrium although the fingertip is moving. Therefore, the relocation in our system is controlled, still in the VF of impedance controller, by a virtual spring with stiffness k connecting the current location of the ith fingertip and $\hat{p}$, an example of fingertip $F1$ relocation is depicted as in Fig. 11, which is equivalent to a fingertip impedance controller superimposed on the original grasp controller.

The stiffness k of the virtual spring is determined by the distance $d_{\hat{p}} \in \mathbb{R}$ between fingertip current location $\hat{p}$ and an empirical parameter $\Gamma \in \mathbb{R}$ as: $k = d_{\hat{p}}\Gamma$. In this way, the fingertip will be smoothly moved towards $\hat{p}$ while keeping the contact on the object, and because $\hat{p}$ is computed in the *HFTS*, the fingertip will not be moved out of the object. By reading tactile feedback and estimating grasp stability online, dependently on how the disturbances changes, the fingertip relocation can stop and start to relocate to another location. An experiment record of virtual spring driven fingertip relocation is shown in Fig. 12, we can see that fingertip $F2$

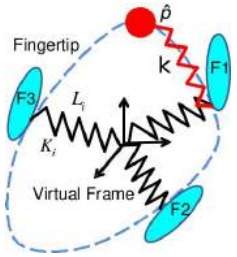


Fig. 11. Virtual spring for fingertip relocation.

stopped moving before the desired location is reached, which is because the grasp is already predicted as stable and then the force adaptation has taken over to maintain the equilibrium.

Fig. 12. The rivella bottle is grasped by the Allegro hand and a human is applying random perturbations on top of it. The red and green points are showing the new locations for fingertip $F1$ and $F2$ computed by Alg. 2 with virtual springs in the virtual frame.

VI. Experiments

We begin this section by first explaining our experiment environment and the system setup. Thereafter, we evaluate the performance of our system in terms of grasp synthesis and grasp adaptation.

In our experiments, the proposed system has been implemented on an Allegro hand equipped with 3 SynTouch[3] tactile sensors on fingertips $F1$, $F2$ and $F3$, as shown in Fig. 8. In order to make the system fully autonomous, we mount the hand on a Kuka LWR arm and employ the algorithm reported in [72] to compute the inverse kinematics of the arm online for reaching the object. 6 objects used in this work, as shown in Fig. 13, are pre-scanned and saved as point clouds. During the grasp execution period, we adopt the OptiTrack[4] real-time motion tracking system to localize the objects, so that the system can localize the object and grasp it. Additionally, the pose and location of the grasp are online updated, so as to also update the pose of *HFTS* during grasp execution when fingertip relocation is required. The tactile based grasp stability estimator is trained in the same way as in [5], in practice, we have observed that when the stability likelihood (E.q 18) is below -130, the object will start to fall, for the sake of safety, we predict a grasp as unstable when the stability likelihood is below -100 to leave a safety margin of 30. Additionally, as described in Sec. IV, the same learned model for stability estimation is also used for our grasp adaptation system to make decisions. For the force control of the hand, we set the initial grasp stiffness $K = (12, 2, 2)$ and use it for the execution of all grasps, as described in Sec. IV, this will be regulated by our learned model Θ through an impedance controller when instability is detected.

Next, we will first evaluate the performance of grasp planning in the pre-grasping phase, and then by executing planned grasps on different objects, we evaluate the performance of the grasp adaptation system by a set of experiments with quantitative results.

A. Grasp Planning

In the pre-grasping phase, as explained in Sec. III, a reachability table with 10^6 hand configurations is prepared by a rejection sampling method: we uniformly sample grasps in the hand joint space and the saved grasps should be self-collision free and have adaptabilities larger than 0.02.

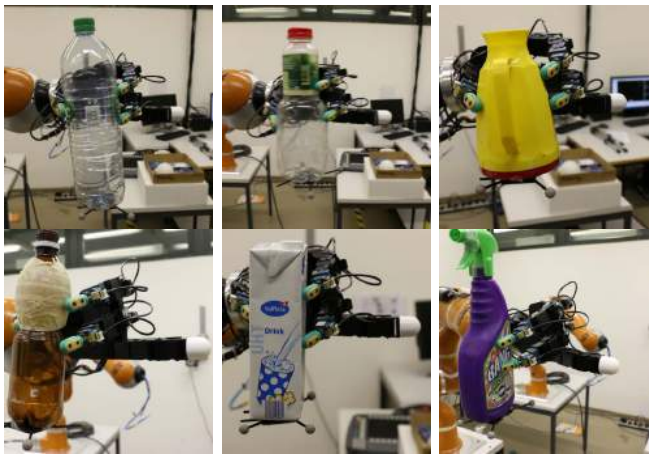


Fig. 13. The best grasps out of 100 grasps optimized by Alg. 1 on 6 test objects: bottle1, bottle2, jug, rivella, milk and spray. Note that as discussed in Sec. III-D
, due to the properties of the objective function used in grasp optimization, the generated grasps should have similar shapes and poses.

When the system is asked to grasp an object, our system uses the object point cloud to compute the *HFTS* and then input it to grasp optimization Algorithm Alg. 1 to synthesize grasps. As described in Sec. III-C, our grasp planner produces grasps by generating both contact locations and hand configurations. Therefore, once a grasp is planned, the grasp execution is done by simply moving the hand to the planned grasping pose and then using the generated hand configuration to execute the grasp by purely position control in the hand joint space. Some example grasps synthesized by our system are shown in Fig. 13.

For evaluating the performance of our grasp planner, we repeat the grasp optimization Algorithm Alg. 1 for each of the test objects for 100 times in simulation, and the evaluation results are summarized into a table shown in Fig. 14. In this test, in order to keep each repetition of the algorithm to have the same number of iterations, we set the $maxIter = 100$ and set $stopCondition(g^i)$ to output always $false$. And in order to evaluate the successful rate of the algorithm, we repeat the algorithm for one more run every time when the produced

[3]http://www.syntouchllc.com/
[4]http://www.naturalpoint.com/optitrack/

grasp is not stable or not collision free, which is checked by executing planned grasp on the object in simulation.

Object(#Units)	#Levels	#Nodes	Avg. Time(s)	Avg. Runs
bottle1(2736)	6	5672	9.71	1.04
bottle2(3102)	6	6321	10.23	1.07
jug(2671)	5	3227	8.26	1.12
rivella(2273)	6	5124	9.36	1.02
milk(2696)	5	3204	7.32	1.03
spray(3207)	6	6926	12.69	1.06

Fig. 14. Evaluation of Alg. 1: #Units: number of fingertip units in $\Phi(\mathcal{P})$, #Levels: number of levels in graph G_Φ (including the top level with only one node), #Nodes: number of nodes in graph G_Φ, Avg. Time(s): average time in seconds for one run of the algorithm, Avg. Runs: average runs of the algorithm to obtain a stable and collision free grasp.

From Fig. 14 we can see that the number of levels of graph G_Φ for all objects are basically between 4 and 5, this small difference implies that our partitioning method produces partition trees that are approximately balanced, in the sense that difference branches in the partition tree terminate at almost the same level. It is worth to note that given almost the same size of fingertip space, for example rivella and milk, the number of nodes can be very different dependently on the number of levels, this is due to when some partitioning branches are terminated, i.e. the number of fingertip units are smaller than 10, all the fingertip units in those branches are directly extended to the next level, which increases the number of nodes in graph G_Φ by a large number. From the average number of runs of Alg. 1 we can see that the algorithm does not guarantee that the produced grasp is always collision free and stable. This is because of the facts that our reachability measure in the system is not capable of checking collisions during grasp optimization, and that the algorithm is randomly initialized and can end up with local optimum. However, as we can see from the results, the number of re-running the algorithm is very small, which means that, in practice, we can just run the algorithm again to plan a new grasp when instability or collision is detected in simulation.

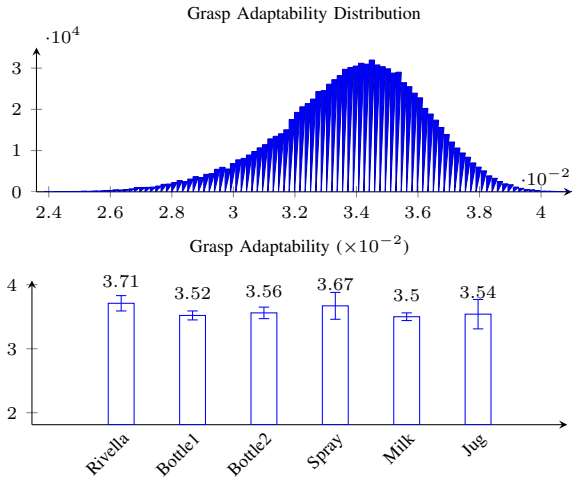


Fig. 15. Adaptability distribution.

The adaptability distribution of 10^6 hand configurations in the reachability lookup table is shown in Fig. 15 with 400 bins, together with the average adaptability of the 100 grasps generated in the evaluation. We can see that our adaptability prioritized reachability lookup table, as shown in Eq. 15, produces good results and the grasp adaptabilities can approximately converge to the same values with different objects.

B. Grasp Adaptation

Once the grasp is executed and contacts are made, our system will enter the post-grasping phase and start to monitor the stability through tactile feedback. In this phase, instead of the position control, the impedance controller is switched on to control the dynamics of the hand in the VF shown in Fig. 8.

1) Experiment Design: In order to quantitatively evaluate the performance of our grasp adaptation system with respect to disturbance rejection, we have designed two sets of experiments: 1) Execute planned grasps on the 6 test objects and fill black pepper beans into them, and then we evaluate what is the maximum weight each grasp can support. And 2) Execute planned grasps on the 6 test objects with different weights, in terms of how many peppers are filled in, we shake the end-effector of the LWR arm with a linearly increasing acceleration in different directions, and then we evaluate what is the maximum acceleration each grasp can withstand. In order to show the advantages of our system, we conduct the same experiments on a system without any grasp adaptation as well as on the system proposed in [5], we show by comparison that our system outperforms both and discuss about the reason behind it. Next, before getting into the details of our experiment, we will first demonstrate a qualitative experiment of grasp adaptation to show the performance of fingertip relocation.

2) An example of fingertip relocation: In this example, we show the efficiency of fingertip relocations in terms of how quickly the real grasp rest lengths follow the desired grasp rest length $\hat{L}^*$ in Eq. 19. For this, once a grasp is planned and executed, we ask a human subject to randomly perturb the object with large force to try to continuously trigger fingertip relocations.

In Fig. 16, 4 example fingertip relocation procedures are depicted by the grasp rest lengths together with their stability estimation. We can see that the fingertip relocation controlled by the virtual spring is able to follow the desired rest lengths by relocating the fingertip to the location computed by Alg. 2. However, we can also observe that the there are always overshots or undershots in this procedure, this is because our fingertip relocation is achieved in the virtual frame by the impedance controller, which is a compliant mechanism and the fingertip rest lengths are also affected by the perturbations. In some cases, e.g., in the second example shown in Fig. 16, the grasp ended up with rest lengths that are very different from the desired one, this is due to the same reason as the example shown in Fig. 12: fingertip relocation is stopped when the grasp is predicted as stable or the system considers force adaptation is able to maintain the equilibrium.

3) Supported Weight Test: In a lot of real world applications, holding an object and carrying it to another place is a

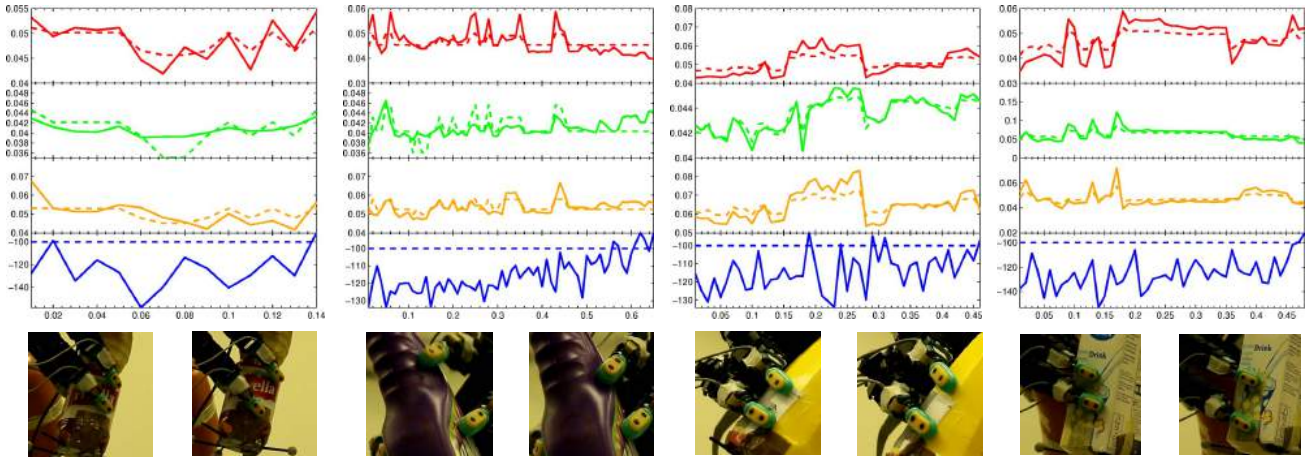


Fig. 16. Fingertip relocation examples are shown for 4 grasps, the red, green and orange lines are showing the real time recorded rest lengths for fingertips $F1$, $F2$ and $F3$ respectively, and the dashed lines are showing the desired rest lengths. The blue lines show the online recored stability likelihood calculated by Eq. 18. The horizontal axis is time in seconds and the rest length is shown in meters. The figures below the diagrams are the grasps before and after the fingertip relocation.

very common task. However, the weight of objects, which can be empty or filled with something, can be very different for the same or different objects, and we may want to fill something into a container when a robot is grasping it. Therefore, it is very important to evaluate the ability of a grasp in term of what is the maximum weight is supports. For this, we select the best grasp from the 100 grasps generated in Sec. VI-A for each object, as shown in Fig. 13, and execute them with the robot. Once a grasp is executed, we adjust the arm configurations to let the hand holds the object perpendicularly to the ground, as shown in Fig. 20, and then we start to gradually fill black pepper beans into the object and record the maximum weight the grasp can support. The maximum weight a grasp can support is determined by 2 criteria: the stability estimator predicts it as unstable for more than 2 seconds or the object drops directly. We repeat this test for each of the grasps for 5 times and summarize the average maximum supported weight for each object into a table shown in Fig. 17. For comparison, we do the same tests with the same grasps for another 2 systems, which are the system proposed in [5] and a system without adaptation.

Object	Weight	Without	With [5]	Improved
bottle1	34	55.1 ± 7.11	153.1 ± 12.31	165.3 ± 13.27
bottle2	39	62.8 ± 6.63	102.3 ± 13.38	121.3 ± 9.91
jug	112	125.3 ± 14.90	147.4 ± 9.62	162.1 ± 13.12
rivella	24	36.0 ± 6.96	76.5 ± 9.4	92.7 ± 7.45
milk	34	63.5 ± 8.20	151.8 ± 7.24	157.4 ± 8.35
spray	63	75.7 ± 7.21	102.2 ± 6.02	121.6 ± 7.15

Fig. 17. The comparison of the supported object weights(Unit:gram). **without**: without grasp adaptation; **with [5]**: with grasp adaptation in [5]; **improved**: the new adaptation approach in this paper.

From the results we can see that the system without adaptation performs the worst, this is simply because that when we increase the weight of an object, we need to apply more force to hold it. We can also see that the system proposed in this work outperforms the system in [5], which does a blind exploration strategy in the tangential plane of contact when fingertip relocation is required, additionally, it assumes only one fingertip can be relocated. Compared with [5] we can observe that the Alg. 2 in this work provides several merits: Firstly, Alg. 2 considers grasp reachability Eq. 13 during the exploration for fingertip relocation, it therefore guarantees that the new location is reachable. Secondly, since the new location is computed in the *HFTS* rather than blind exploration, we can make sure the new location does exist in reality, the potential risk of blind exploration is shown in Fig. 18. Last but not least, Alg. 2 computes fingertip relocation for 2 fingertips, which can be easily extended to multiple fingertips, it therefore provides smaller optimization residual of grasp rest lengths. All these mentioned merits can decrease the relocation errors throughout the whole process of grasping, and it is worth noticing that small errors can be accumulated into big errors. Therefore, it is not difficult to understand why the proposed system can support more weights than the other two systems.

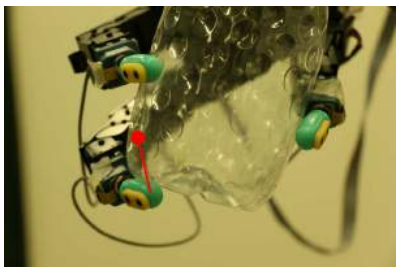


Fig. 18. The risk of relocating a fingertip to an non-existing position [5] can be removed by the proposed system using visual information represented by *HFTS*.

A quantitative evaluation between the proposed system and the system in [5] has been conducted with respect to optimization residual. In this evaluation, we first execute the grasp in simulation, and then we intentionally trigger the fingertip relocation system by sending desired rest lengths randomly sampled around the current values within a ball of radius $0.02m$. This test is conducted on both systems and the result is shown in Fig. 19. We can see that due to the object

shape constraint, both systems can not provide zero residuals. However, the proposed system performs much better in this case and is less object shape dependent. As aforementioned, this smaller optimization residual results in less accumulated errors in the whole process of grasp execution.

An example of the supported weight test for rivella bottle is shown in Fig. 20. We can see that in the beginning when the object is not too heavy, the likelihood $p(\hat{(g)}|\Theta)$ is larger than -100 and the grasp stiffness is a constant value. However, while the weight increases, the grasp becomes unstable and the adaptation system starts to adapt grasp stiffness, and we can see, as expected, that the grasp stiffness is generally increasing. After the weight has been increased to a certain level, the adaptation system increases grasp stiffness rapidly and the stability drops rapidly, and then the system figured out that stiffness adaptation is not enough now, a fingertip relocation has been triggered and the fingertip $F1$ has been relocated. Observe that, the grasp stiffness has been decreased after this relocation since the new grasp is better in this situation. Thereafter, we can see that the grasp stiffness is still increasing while the weight increases, another fingertip relocation of $F2$ has been applied to keep the grasp stable when there is another rapid drop of grasp stability.

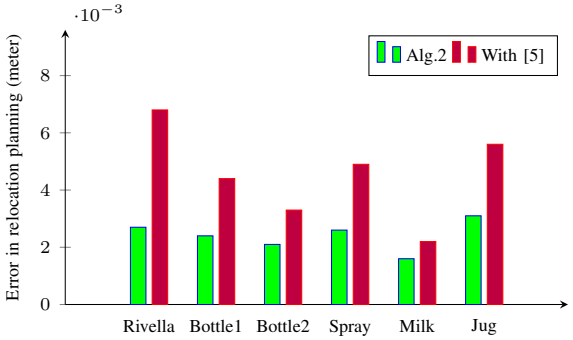


Fig. 19. The comparison of fingertip relocation optimization residual, 10^5 desired rest lengths are randomly sample around the current rest length within a ball of radius $0.02m$.

4) Shaking Test: External disturbances can occur during a grasp execution, e.g., a grasp can be perturbed by a human hand or another robot when some collision happens, and this can potentially cause the grasp to drop the object. Therefore, the ability of withstanding external disturbances is crucial for grasps. In this section, in order to quantitatively evaluate the system performance with respect external disturbances, we have designed a grasp shaking test with a setup shown in Fig. 21. In this evaluation, we still use the best grasp out of the 100 grasps generated in Sec. VI-A for each object, and once a grasp is executed, we change the arm configuration to a fixed configuration shown in Fig. 21, such that the opening direction of the hand is perpendicular to the ground. Thereafter, we start to shake the robot hand in either vertical or horizontal directions while gradually increasing the linear acceleration rate from $2m/s^2$ to $8m/s^2$. In this test, the shaking magnitude is limited to $10cm$ in either directions, which means that the hand is accelerating in the first $5cm$ and decelerating in the second $5cm$.

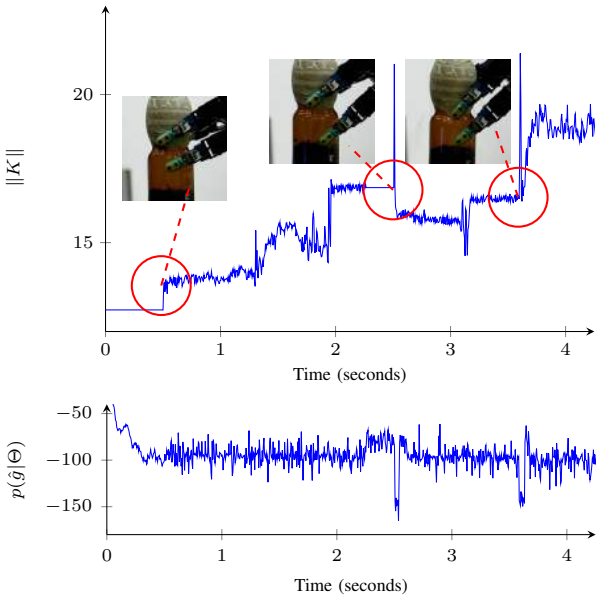


Fig. 20. A record of supported weight test of a grasp on rivella bottle. Upper: The norm of grasp stiffness and fingertip relocation. Lower: Likelihood for grasp stability estimation defined in Eq. 18.

$J0$	$J1$	$J2$	$J3$	$J4$	$J5$	$J6$
$-30°$	$30°$	$2°$	$-60°$	$-20°$	$0°$	$-60°$

Init. $K = (K_x, K_y, K_z)$	Horizontal Acc.	Vertical Acc.
$(12, 2, 2)$	$2m/s^2$ – $8m/s^2$	$2m/s^2$ – $8m/s^2$

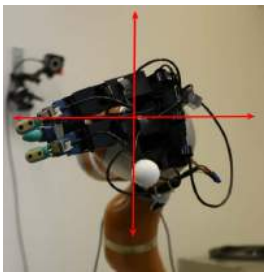


Fig. 21. The setup of grasp shaking test, in which the arm shakes each grasp in horizontal and vertical directions. When shaking horizontally, the shaking direction is fixed to be perpendicular to the palm.

Similarly to the the supported weight test, we evaluate, each grasp by measuring what is the maximum acceleration it can withstand, and the criterion is the same: the maximum acceleration is recorded when the grasp is predicted as unstable for more that 2 seconds or the object drops directly. In order to evaluate this under the uncertainty of object weights, we conduct the shaking test, in both directions separately, on each object by filling in $10g$, $20g$, $30g$, $40g$ and $50g$ black pepper beans and repeat each test for 5 times. For comparison, we also conduct the same test for a system without adaptation and the system of [5]. The experiment results are summarized in Fig. 22, Note that when if the maximum acceleration rate is $8m/s^2$, it means the object has been always stable during the test, while if the maximum acceleration rate is $2m/s^2$, it means the object can not withstand any shaking or even can not be picked up.

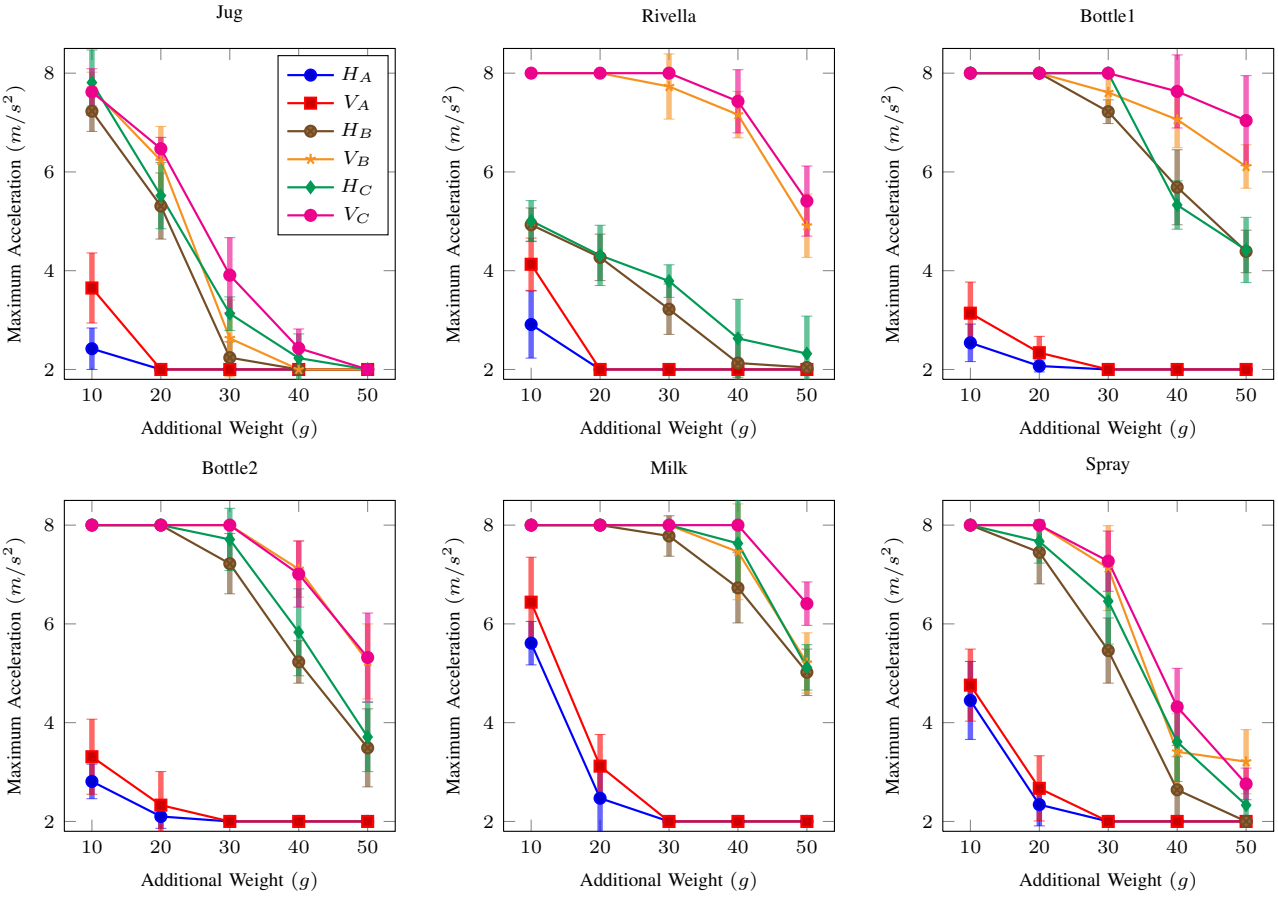


Fig. 22. Results of shaking tests on grasps shown in Fig. 13. In the legend, H and V refer to horizontal shaking test and vertical shaking test respectively. A, B and C refer to 3 grasp strategies: grasp without adaptation, grasp adaptation in [5] and the grasp adaptation proposed in this paper.

Object	Avg. Duration(ms)	Avg. Improvement	Avg. Comp. Time(ms)	Avg. Err.(m)	Avg. # Nodes	Relocation/Adaptation(%)
bottle1	261.2	66.21	30.7	0.0074	279.2	0.23
bottle2	320.1	75.17	32.1	0.0062	221.7	0.16
jug	414.4	70.72	19.4	0.0042	140.4	0.12
rivella	447.9	52.39	38.6	0.0045	194.1	0.27
milk	392.6	47.11	24.9	0.0057	137.6	0.45
spray	502.7	57.26	29.2	0.0068	197.7	0.31

Fig. 23. System statistics of the grasp shaking tests. The statistics recored and shown in the table are (from left to right): Average relocation duration until fingertip is fixed. Average stability likelihood improvement after fingertip relocation. Average computation time of Alg. 2 for each computation. Average errors between achieved rest lengths and the rest lengths computed by Alg. 2. Average number of nodes explored in Alg. 2. The ratio of fingertip relocations needed when grasp adaptation is required.

From the results we can see that grasps without adaptation enabled are very weak against perturbations, which is intuitive since external perturbations may require more force to counteract than needed by the weight supported test. We can observe that the proposed system in this work performs better than [5] due to the same reason discussed in supported weight test, and more importantly, since the grasp has more uncertainties in the shaking test, such as center of mass of the object due to the black pepper beans, the accumulated errors can cause more problems. The results also implies that the grasps can withstand more acceleration rate in vertical shaking tests, this is because the filled pepper beans are at the bottom of the containers, and this causes more torque to be required to stabilize a grasp during horizontal shaking.

Some additional quantitative results for the proposed system are recorded as shown in Fig. 23. From the result we can see that the average computation time of Alg. 2 is between $20ms$ to $40ms$, which means that our adaptation system can work at between $25Hz$ to $50Hz$ online, and this is sufficient to support the computation of fingertip relocation. The average number of nodes for exploration shows that the pruning works very well in the algorithm and we only need to check several hundred nodes to decide where to relocate a fingertip. Note that the computation time and number of explored nodes are heavily dependent on the connectivity of graph G_Φ, less node in the graph does not mean less computation time.

It is worth to note that the average errors between achieved rest lengths and the rest lengths computed by Alg. 2 is not

a negligible, and this reminds us about how the accumulated error can cause big problems. Consider that if we have a large optimization residual, the grasp rest lengths achieved by the system can be quite different from the desired value. The last column implies that in most of cases, grasp force adaptation is enough to stabilize a grasp, this can also be observed in Fig. 20, in which only 2 fingertip relocations were needed over more than 1000 force adaptations.

VII. Conclusion

References

[1] Y. Bekiroglu, D. Song, L. Wang, and D. Kragic, "A probabilistic framework for task-oriented grasp stability assessment," in *IEEE ICRA*, 2013.

[2] K. Hang, J. A. Stork, F. T. Pokorny, and D. Kragic, "Combinatorial optimization for hierarchical contact-level grasping," in *IEEE ICRA*, 2014.

[3] C. Borst, M. Fischer, and G. Hirzinger, "Calculating hand configurations for precision and pinch grasps," in *IEEE/RSJ IROS*, 2002.

[4] K. Tahara, S. Arimoto, and M. Yoshida, "Dynamic object manipulation using a virtual frame by a triple soft-fingered robotic hand," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, May 2010, pp. 4322–4327.

[5] M. Li, Y. Bekiroglu, D. Kragic, and A. Billard, "Learning of grasp adaptation through experience and tactile sensing," in *IEEE/RSJ IROS*, 2014.

[6] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis – a survey," *Robotics, IEEE Transactions on*, vol. 30, no. 2, pp. 289–309, 2014.

[7] O. Khatib, S. Quinlan, and D. Williams, "Robot planning and control," *Robotics and Autonomous Systems*, vol. 21, no. 3, pp. 249 – 261, 1997, critical Issues in Robotics. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889096000784

[8] K. Shimoga, "Robot grasp synthesis algorithms: A survey," *The International Journal of Robotics Research*, vol. 15, no. 3, pp. 230–266, 1996.

[9] A. Sahbani, S. El-Khoury, and P. Bidaud, "An Overview of 3D Object Grasp Synthesis Algorithms," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 326–336, 2011.

[10] T. Yoshikawa, "Multifingered robot hands: Control for grasping and manipulation," *Annual Reviews in Control*, vol. 34, no. 2, pp. 199 – 208, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1367578810000416

[11] J. Romano, K. Hsiao, G. Niemeyer, S. Chitta, and K. Kuchenbecker, "Human-inspired robotic grasp control with tactile sensing," *Robotics, IEEE Transactions on*, vol. 27, no. 6, pp. 1067–1079, Dec 2011.

[12] M. R. Cutkosky and R. D. Howe, "Dextrous robot hands," S. T. Venkataraman and T. Iberall, Eds. New York, NY, USA: Springer-Verlag New York, Inc., 1990, ch. Human Grasp Choice and Robotic Grasp Analysis, pp. 5–31.

[13] K. Hang, J. A. Stork, and D. Kragic, "Hierarchical fingertip space for multi-fingered precision grasping," in *IEEE/RSJ IROS*, Chicago, US, 2014.

[14] Z. Gao, "Active disturbance rejection control: a paradigm shift in feedback control system design," in *American Control Conference, 2006*, June 2006, pp. 7 pp.–.

[15] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: a survey," *Cognitive Processing*, vol. 12, no. 4, pp. 319–340, 2011. [Online]. Available: http://dx.doi.org/10.1007/s10339-011-0404-1

[16] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469 – 483, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889008001772

[17] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *IEEE ICRA*, 2000.

[18] C. Borst, M. Fischer, and G. Hirzinger, "Grasping the dice by dicing the grasp," in *IEEE/RSJ IROS*, 2003.

[19] B. Mirtich and J. Canny, "Easily computable optimum grasps in 2-d and 3-d," in *In IEEE International Conference on Robotics and Automation*, 1994, pp. 739–747.

[20] J.-P. Saut and D. Sidobre, "Efficient models for grasp planning with a multi-fingered hand," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 347 – 357, 2012.

[21] C. Rosales, J. Porta, and L. Ros, "Grasp optimization under specific contact constraints," *Robotics, IEEE Transactions on*, vol. 29, no. 3, pp. 746–757, June 2013.

[22] *Three-Finger Precision Grasp on Incomplete 3D Point Clouds*, Hong Kong, China, 2014.

[23] C. Ferrari and J. Canny, "Planning optimal grasps," in *IEEE ICRA*, 1992.

[24] C. Rosales, J. Porta, R. Suarez, and L. Ros, "Finding all valid hand configurations for a given precision grasp," in *IEEE ICRA*, 2008.

[25] K. Hang, F. T. Pokorny, and D. Kragic, "Friction coefficients and grasp synthesis," in *IEEE/RSJ IROS*, Tokyo, Japan, 2013.

[26] M. Roa and R. Suarez, "Computation of independent contact regions for grasping 3-d objects," *Robotics, IEEE Transactions on*, vol. 25, no. 4, pp. 839–850, 2009.

[27] K. Hertkorn, M. Roa, and B. Ch, "Planning in-hand object manipulation with multifingered hands considering task constraints," in *IEEE ICRA*, 2013.

[28] C. Borst, M. Fischer, and G. Hirzinger, "Grasp planning: how to choose a suitable task wrench space," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 1, April 2004, pp. 319–325 Vol.1.

[29] L. Han and J. Trinkle, "Dexterous manipulation by rolling and finger gaiting," in *IEEE ICRA*, 1998.

[30] J.-P. Saut, A. Sahbani, S. El-Khoury, and V. Perdereau, "Dexterous manipulation planning using probabilistic roadmaps in continuous grasp subspaces," in *IEEE/RSJ IROS*, 2007.

[31] J. Aleotti and S. Caselli, "A 3d shape segmentation approach for robot grasping by parts," *IEEE RAS*, vol. 60(3), pp. 358–366, 2012.

[32] M. Przybylski, T. Asfour, and R. Dillmann, "Planning grasps for robotic hands using a novel object representation based on the medial axis transform," in *IEEE/RSJ IROS*, 2011.

[33] N. Vahrenkamp, M. Przybylski, T. Asfour, and R. Dillmann, "Bimanual grasp planning," in *IEEE-RAS Humanoids*, 2011, pp. 493–499.

[34] K. Huebner, "BADGrA toolbox for box-based approximation, decomposition and GRasping," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 367 – 376, 2012.

[35] R. Pelossof, A. Miller, P. Allen, and T. Jebara, "An svm learning approach to robotic grasping," in *IEEE ICRA*, vol. 4, 2004, pp. 3512–3518 Vol.4.

[36] G. Biegelbauer and M. Vincze, "Efficient 3d object detection by fitting superquadrics to range image data for robot's object manipulation," in *IEEE ICRA*, 2007.

[37] T. T. Cocias, S. M. Grigorescu, and F. Moldoveanu, "Multiple-superquadrics based object surface estimation for grasping in service robotics," in *Optimization of Electrical and Electronic Equipment (OPTIM)*, 2012, pp. 1471–1477.

[38] D. Berenson, R. Diankov, K. Nishiwaki, S. Kagami, and J. Kuffner, "Grasp planning in complex scenes," in *IEEE-RAS Humanoids*, 2007.

[39] J. A. Stork, F. T. Pokorny, and D. Kragic, "Integrated motion and clasp planning with virtual linking," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'2013)*, Tokyo, Japan, 2013.

[40] F. T. Pokorny, J. A. Stork, and D. Kragic, "Grasping objects with holes: A topological approach," in *IEEE ICRA*, 2013.

[41] D. Song, C. Ek, K. Huebner, and D. Kragic, "Embodiment-specific representation of robot grasping using graphical models and latent-space discretization," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, Sept 2011, pp. 980–986.

[42] C. Ek, D. Song, K. Huebner, and D. Kragic, "Task modeling in imitation learning using latent variable models," in *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*, Dec 2010, pp. 548–553.

[43] C. Zhang, D. Song, and H. Kjellstrom, "Contextual modeling with labeled multi-lda," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'2013)*, Tokyo, Japan, 2013.

[44] C. Xiong and Y. Xiong, "Neural-network based force planning for multifingered grasp," *Robotics and Autonomous Systems*, vol. 21, no. 4, pp. 365 – 375, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889097000201

[45] Y. Xia, J. Wang, and L.-M. Fok, "Grasping-force optimization for multifingered robotic hands using a recurrent neural network," *Robotics and Automation, IEEE Transactions on*, vol. 20, no. 3, pp. 549–554, June 2004.

[46] T. Yoshikawa, "Multifingered robot hands: Control for grasping and manipulation," *Annual Reviews in Control*, vol. 34, no. 2, pp. 199 – 208, 2010.

[47] Z. Li, P. Hsu, and S. Sastry, "Grasping and coordinated manipulation by a multifingered robot hand." *The International Journal of Robotics Research*, vol. 8, no. 4, pp. 33–50, 1989.

[48] T. Yoshikawa and X.-Z. Zheng, "Coordinated dynamic hybrid position/force control for multiple robot manipulators handling one constrained object," *The International Journal of Robotics Research*, vol. 12, no. 3, pp. 219–230, 1993.

[49] M. Buss, H. Hashimoto, and J. Moore, "Dextrous hand grasping force optimization," *Robotics and Automation, IEEE Transactions on*, vol. 12, no. 3, pp. 406–418, Jun 1996.

[50] T. Takahashi, T. Tsuboi, T. Kishida, Y. Kawanami, S. Shimizu, M. Iribe, T. Fukushima, and M. Fujita, "Adaptive grasping by multi fingered hand with tactile sensor based on robust force and position control," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, May 2008, pp. 264–271.

[51] S. A. Schneider and R. H. Cannon, "Object impedance control for cooperative manipulation: theory and experimental results," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 3, pp. 383–394, 1992.

[52] H. Liu and G. Hirzinger, "Cartesian impedance control for the DLR hand," in *Proceedings of International Conference on Intelligent Robots and Systems (IROS), 1999.*

[53] K. Tahara, K. Maruta, A. Kawamura, and M. Yamamoto, "Externally sensorless dynamic regrasping and manipulation by a triple-fingered robotic hand with torsional fingertip joints," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2012.*

[54] M. Li, H. Yin, K. Tahara, and A. Billard, "Learning object-level impedance control for robust grasping and dexterous manipulation," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2014.*

[55] K. J. AAström and T. Hägglund, *PID Controllers: Theory, Design, and Tuning*, 2nd ed. Instrument Society of America, Research Triangle Park, NC, 1995.

[56] I. Havoutis, C. Semini, J. Buchli, and D. Caldwell, "Quadrupedal trotting with active compliance," in *Mechatronics (ICM), 2013 IEEE International Conference on*, Feb 2013, pp. 610–616.

[57] J. G. D. Karssen and M. Wisse, "Running with improved disturbance rejection by using non-linear leg springs." *I. J. Robotic Res.*, vol. 30, no. 13, pp. 1585–1595, 2011. [Online]. Available: http://dblp.uni-trier.de/db/journals/ijrr/ijrr30.html#KarssenW11

[58] J. J. Alcaraz-Jimnez, M. Missura, H. M. Barber, and S. Behnke, "Lateral disturbance rejection for the nao robot." in *RoboCup*, ser. Lecture Notes in Computer Science, X. Chen, P. Stone, L. E. Sucar, and T. van der Zant, Eds., vol. 7500. Springer, 2012, pp. 1–12. [Online]. Available: http://dblp.uni-trier.de/db/conf/robocup/robocup2012.html#Alcaraz-JimenezMBB12

[59] S. Hyon and G. Cheng, "Disturbance rejection for biped humanoids," in *Robotics and Automation, 2007 IEEE International Conference on*. IEEE, 2007, pp. 2668–2675.

[60] J. Rebula, F. Canas, J. Pratt, and A. Goswami, "Learning capture points for humanoid push recovery," in *Humanoid Robots, 2007 7th IEEE-RAS International Conference on*, Nov 2007, pp. 65–72.

[61] A. Rodriguez, M. T. Mason, and S. Ferry, "From caging to grasping," *The International Journal of Robotics Research*, vol. 31, no. 7, pp. 886–900, 2012.

[62] S. Koziel, D. Ciaurri, and L. Leifsson, "Surrogate-based methods," in *Computational Optimization, Methods and Algorithms*, ser. Studies in Computational Intelligence, S. Koziel and X.-S. Yang, Eds. Springer Berlin Heidelberg, 2011, vol. 356, pp. 33–59. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-20859-1_3

[63] C. Walshaw, "Multilevel refinement for combinatorial optimisation problems," *Annals of Operations Research*, vol. 131, no. 1-4, pp. 325–372, 2004.

[64] F. T. Pokorny and D. Kragic, "Classical grasp quality evaluation: New theory and algorithms," in *IEEE/RSJ IROS*, 2013.

[65] N. Sommer, L. Miao, and A. Billard, "Bimanual compliant tactile exploration for grasping unknown objects," in *Proceedings of International Conference on Robotics and Automation (ICRA), 2014.*

[66] O. Williams and A. Fitzgibbon, "Gaussian process implicit surfaces," in *Gaussian Proc. in Practice*, 2007.

[67] F. L. Bookstein, "Principal warps: thin-plate splines and the decomposition of deformations," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 11, no. 6, pp. 567–585, Jun 1989.

[68] T. Yoshikawa, "Manipulability of robotic mechanisms," *IJRR*, 1985.

[69] L. Guilamo, J. Kuffner, K. Nishiwaki, and S. Kagami, "Efficient prioritized inverse kinematic solutions for redundant manipulators," in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, Aug 2005, pp. 3921–3926.

[70] Y. Aydin and M. Nakajima, "Database guided computer animation of human grasping using forward and inverse kinematics." *Computers & Graphics*, vol. 23, pp. 145–154, 1999.

[71] M. Powell, "A direct search optimization method that models the objective and constraint functions by linear interpolation," in *Advances in Optimization and Numerical Analysis*, ser. Mathematics and Its Applications, S. Gomez and J.-P. Hennart, Eds. Springer Netherlands, 1994, vol. 275, pp. 51–67. [Online]. Available: http://dx.doi.org/10.1007/978-94-015-8330-5_4

[72] A. Shukla and A. Billard, "Coupled dynamical system based arm-hand grasping model for learning fast adaptation strategies," *Robot. Auton. Syst.*, vol. 60, no. 3, Mar. 2012.

Bibliography

Fox E, Sudderth E, Jordan M, Willsky A (2009) Sharing features among dynamical systems with beta processes. In: NIPS, vol 22, pp 549–557

Ge SS, Lee TH, Ren SX (2001) Adaptive friction compensation of servo mechanisms. International Journal of Systems Science 32:523–532

Hughes M, Fox E, Sudderth E (2012) Effective split-merge monte carlo methods for nonparametric models of sequential data. In: NIPS, vol 25, pp 1304–1312

Li M, Bekiroglu Y, Kragic D, Billard A (2014a) Learning of grasp adaptation through experience and tactile sensing. In: IEEE/RSJ IROS

Li M, Yin H, Tahara K, Billard A (2014b) Learning object-level impedance control for robust grasping and dexterous manipulation. In: Proceedings of the International Conference on Robotics and Automation (ICRA), 2014. (submitted)

Pais AL, Umezawa K, Nakamura Y, Billard A (2014) Task parametrization using continuous constraints extracted from human demonstrations. Submitted

Shukla A, Billard A (2012) Coupled dynamical system based arm-âĂŞhand grasping model for learning fast adaptation strategies. Robotics and Autonomous Systems 60(3):424 – 440