Foundations and Trends® in
sample
Vol. 1, No 2 (2010) 1–54

DOI: 10.1561/2300000012

now
the essence of knowledge

Tactile Guidance for Policy Adaptation

Brenna D. Argall, Eric L. Sauser and Aude G. Billard[1]

[1] *Ecole Polytechnique Fédérale de Lausanne (EPFL)Lausanne1015, Switzerland, brennadee.argall@epfl.ch*

Abstract

Demonstration learning is a powerful and practical technique to develop robot behaviors. Even so, development remains a challenge and possible demonstration limitations, for example correspondence issues between the robot and demonstrator, can degrade policy performance. This work presents an approach for policy improvement through a tactile interface located on the body of the robot. We introduce the *Tactile Policy Correction (TPC)* algorithm, that employs tactile feedback for the *refinement* of a demonstrated policy, as well as its *reuse* for the development of other policies. The TPC algorithm is validated on humanoid robot performing grasp positioning tasks. The performance of the demonstrated policy is found to improve with tactile corrections. Tactile guidance also is shown to enable the development of policies able to successfully execute novel, undemonstrated, tasks. We further show that different modalities, namely teleoperation and tactile control, provide information about allowable variability in the target behavior in different areas of the state space.

Contents

1

Introduction

The realization of physical movement is fundamental to many robotics applications. Whether operating in industrial and laboratory settings, or within general society, physically embodied robots typically are tasked with the execution of physical actions, thus requiring an algorithm for motion control. Over the years a variety of approaches for motion control have been proposed, with many resulting in impressive robot capabilities. The development of control paradigms becomes increasingly difficult however as robot and domain complexities grow, for example with high degree-of-freedom manipulators or interactions with compliant objects. Often traditional approaches that define explicit mathematical models of the world, and from these derive rules for control, struggle to scale with increasing complexity. Moreover, the development of a control paradigm for any robot platform is confounded by difficulties such as noisy sensors and inaccurate actuation.

In the face of such challenges, to develop robust control algorithms typically requires a significant measure of expertise and effort from the developer. The advancement of techniques that reduce the demands placed on a developer therefore are desirable. We introduce in this article an approach to policy development in which corrections provided

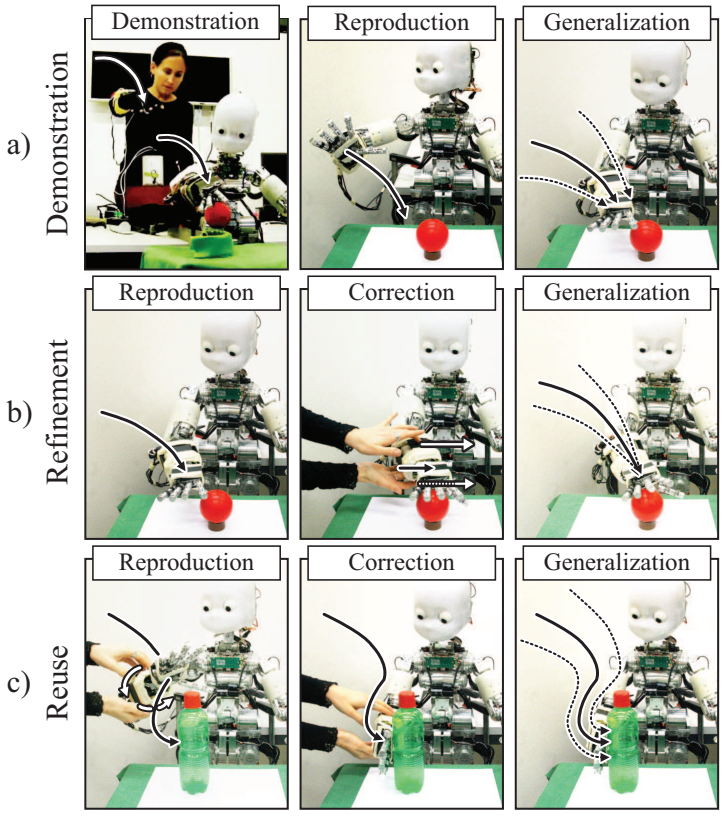

Fig. 1.1 Our approach of a) task demonstration, followed by tactile correction of the learned policy for b) refinement of the demonstrated behavior and c) its reuse in the development of other policies. Black solid arrows indicate demonstrated or corrected executions, black dashed arrows generalization executions and white arrows human hand movement.

by a teacher through a tactile interface are used to adapt and improve a policy. Our *Tactile Policy Correction (TPC)* algorithm initially derives a policy via *Learning from Demonstration (LfD)* techniques (Fig. 1.1a). Under LfD, a robot learner generalizes a policy from data recorded during the execution of a target behavior by a task expert. Our approach then has a human teacher provide policy corrections through a tactile interface located on the body of the robot. The corrections indicate relative adjustments to the robot pose, and thus to the policy predictions. The teacher provides corrections in order to accomplish one of two goals, and how corrections are incorporated into the policy differs

for each. The first goal is to *refine* a policy during execution, and thus to improve its performance based on execution experience (Fig. 1.1b). The second goal is to assist in policy *reuse*, by guiding an existing policy towards accomplishing a different task (Fig. 1.1c).

We validate our approach on a humanoid robot performing end-effector positioning tasks. We show that policies produced under our policy derivation technique are flexible with respect to variability seen between the teacher demonstrations, and furthermore that different teaching modalities (i.e. task demonstration, tactile correction) provide information about acceptable execution variability within different areas of the state space. The performance of a policy learned from demonstration is shown to improve after *refinement* through tactile corrections. Successful policy *reuse* also is validated. Through tactile guidance, executions with existing policies are iteratively adjusted towards producing new behaviors, with the result of policies able to execute alternate, undemonstrated, tasks. Tactile corrections thus *enable* the development of new policies, bootstrapped on the reuse of a policy learned from demonstration.

The remainder of this chapter reports on the related literature that supports this work. Chapter 2 introduces the TPC algorithm and presents our implementation in detail. Experimental setup and results are reported in Chapter 3. A discussion of our approach and findings are provided in Chapter 4, followed by concluding remarks.

1.1 Background and Motivation

We begin with a discussion of policy development under *Learning from Demonstration (LfD)*, followed by existing approaches to policy refinement and reuse within LfD.

1.1.1 Learning from Demonstration

Under LfD, teacher executions of a desired behavior are recorded and a policy is derived from the resultant dataset. LfD has seen success in a variety of robotics applications, and has the attractive characteristics of being an intuitive means for human teacher to robot learner knowledge transfer, as well as being an accessible policy development technique for

those who are not robotics-experts. There are many design decisions to consider when building an LfD system. These range from who executes the demonstrations and how they are recorded, to the technique used for policy derivation. Here we overview only those decisions specific to our particular system, and refer the reader to Argall et al. (2009) and Billard et al. (2008) for a full review of robot LfD.

When recording and executing demonstrations the issue of *correspondence* is key, where teacher demonstrations do not directly map to the robot learner due to differences in sensing or motion (Nehaniv and Dautenhahn, 2002). Correspondence issues are minimized when the learner records directly from its own sensors while under the control of the teacher. For example, under *teleoperation* the teacher remotely controls the robot platform (e.g. Sweeney and Grupen (2007)), while under *kinesthetic control* the teacher touches the robot to guide the motion (e.g. Calinon and Billard (2007)). Teleoperation requires an interface for the direct control of all degrees of freedom on the robot. By contrast, kinesthetic teaching requires a (passive or active) responsiveness to human touch, for example back-drivable motors or force-torque sensing in the joints. Both techniques are employed in our work.

Many approaches exist within LfD to derive a policy from the demonstration data (Argall et al., 2009), the most popular of which either directly approximate the underlying function mapping observations to actions, or approximate a state transition model and then derive a policy using techniques such as Reinforcement Learning (Sutton and Barto, 1998). Our work derives a policy under a variant of the first approach, where probabilistic regression techniques are used to predict a target robot pose based on world state, and a controller external to the algorithm selects an action able to accomplish this target pose. Our reason for splitting policy prediction into these two steps is tied to the mechanism by which the algorithm responds to tactile feedback (discussed in Sec. 2.1).

1.1.2 Policy Refinement and Reuse

Even with the advantages secured through demonstration, policy development typically is still non-trivial. To have a robot learn from its

execution performance, or *experience*, therefore is a valuable policy improvement tool for any development technique. Within the context of LfD specifically, execution experience can be used to overcome limitations in the demonstration dataset. One possible limitation is dataset sparsity, since demonstration from every world state is infeasible in all but the simplest domains. Other limitations include poor correspondence between the teacher and learner or deficiencies in the teacher, who may in fact provide suboptimal or ambiguous demonstrations. Here we consider policy *refinement* and policy *reuse* as two techniques to assist the development process, or equivalently to reduce the strain on the policy developer.

Within demonstration learning, a variety of approaches incorporate information gathered from experience in order to *refine* a policy. For example, execution experience is used to update reward-determined state values (Guenter et al., 2007; Kober and Peters, 2009; Stolle and Atkeson, 2007) and learned state transition models (Abbeel and Ng, 2005; Bagnell and Schneider, 2001). Other approaches provide more demonstration data, driven by teacher-initiated demonstrations (Calinon and Billard, 2007) as well as by learner requests for more data (Chernova and Veloso, 2008; Grollman and Jenkins, 2007). In this work, we also provide more data, but using a different control mechanism than during the initial teacher demonstrations; specifically, teleoperation is used for the initial demonstration data, and a form of hybrid kinesthetic control when producing the refinement data.

Policy *reuse* under LfD occurs most frequently with behavior primitives, or simpler policies that contribute to the execution of a more complex policy. Hand-coded behavior primitives are used within tasks learned from demonstration (Nicolescu and Mataric, 2003), demonstrated primitives are combined into a new policy by a human (Saunders et al., 2006) or automatically by the learning algorithm (Argall, 2009), and demonstrated tasks are decomposed into a library of primitives (Bentivegna, 2004). The focus of our approach is instead on adapting an existing policy to accomplish a *different* task, rather than incorporating the existing behavior as a subcomponent of a larger task.

1.1.3 Tactile Corrections

To enable policy refinement and reuse, the approach taken in this work is to provide *corrections* on a policy execution. Corrections have the advantage of providing guidance on a more suitable alternate prediction for the policy, instead of requiring that this be inferred from an indication of prediction quality, as state reward does for example. Having directed feedback becomes particularly relevant when guiding a policy towards accomplishing a novel behavior.

Within LfD policy correction has seen limited attention, and most examples consider a human teacher selecting the correct prediction from a discrete set of actions with significant time duration (Chernova and Veloso, 2008; Nicolescu and Mataric, 2003). The target application domain for our work however has policies making continuous-valued predictions at a rapid rate, and both features complicate the individual selection of a single alternate prediction to serve as the correction. To address these challenges, we translate feedback from a tactile sensor into continuous-valued modifications of the current pose online, as the robot executes. In contrast to other work with continuous-valued corrections (Argall, 2009), we offer corrective feedback online, instead of post-execution, and through a tactile interface, instead of a high-level computational language.

We posit that tactile feedback furthers many of the strengths of demonstration-based learning. Namely, humans already use touch to instruct other humans in certain contexts; for example when demonstrating a motion, like a tennis swing, that requires a particular position trajectory. To augment demonstration learning with tactile feedback therefore is one natural extension to the idea of teaching robots as humans teach other humans. Demonstration-based policy development also is accessible to those who are not robotics experts, and possibly operating robots outside of laboratory or industrial settings. Here the detection of tactile interactions can be critical for safe robot operation around humans, and so tactile sensing gains importance on a very fundamental level. These tactile sensing capabilities might then be additionally exploited, to transfer knowledge from human to robot for the purpose of behavior development.

Within the field of robot learning (including but not restricted to LfD), only a handful of works utilize human touch for the development of robot behaviors. For example, tactile feedback is detected in order to minimize resistance to movement during demonstration with an industrial arm (Grunwald et al., 2003), and to minimize the support forces provided by a teacher during humanoid behavior learning (Minato et al., 2007). Tactile interactions between a robotic pet-surrogate and elderly patients also are mapped to reward signals, that are used within a Reinforcement Learning paradigm to adapt behavior selection (Wada and Shibata, 2007).

1.2 Our Approach

In summary, the approach presented in this paper employs *tactile corrections* to modify a policy learned through demonstration, for the purpose of both policy *refinement* and policy *reuse*.

Our target application domain is low-level motion control for high degree-of-freedom (DoF) robots. To specify a target behavior for each joint is complicated, and systems typically are under-constrained, resulting in for example many joint configurations mapping to a single end-effector pose. The ability to exploit previously learned domain knowledge for the development of new policy behaviors, i.e. policy reuse, thus is advantageous. Performance might suffer however if the reused policy provides only an approximation to the new target behavior. Moreover, while the use of demonstration for policy development is practical for many reasons, it is limited by the interface controlling the demonstration, the quality of which furthermore frequently degrades as the degrees of freedom to control increase. We aim to overcome policy deficiencies through refinement.

To accomplish both refinement and reuse, the policy incorporates new behavior examples. Instead of producing the examples from teacher demonstration however (Calinon and Billard, 2007; Chernova and Veloso, 2008; Grollman and Jenkins, 2007), which would be unable to improve upon limitations like a poor demonstration interface, we have the student respond online to corrections indicated by a teacher and treat the resultant trajectory as new training data. Providing explicit

corrections has been seldom used within the LfD paradigm (Chernova and Veloso, 2008; Nicolescu and Mataric, 2003), especially when the corrections are continuous-valued and rapidly sampled (Argall, 2009).

We provide corrections through a tactile interface. In addition to being a technique that is relatively unaddressed to date within the robot learning literature in general (Minato et al., 2007; Wada and Shibata, 2007), and LfD literature in particular (Grunwald et al., 2003), we argue that information transfer through human touch is a natural extension of human demonstration, as an intuitive and effective mechanism for the transfer of knowledge from human to robot.

2

The Tactile Policy Correction Algorithm

We introduce *Tactile Policy Correction (TPC)* as an algorithm for the refinement and reuse of motion policies, accomplished via tactile feedback from a human teacher (Argall et al., 2010). A policy initially is derived from demonstrations of a task by a teacher. Through tactile corrections, the policy then either is refined to better perform the demonstrated task, or modified to accomplish an undemonstrated task and thus reused in the development of a new policy. An overview of the algorithm flow is provided in Figure 2.1, and pseudo-code for this approach in Algorithm 1.

We formally define the world to consist of actions $\boldsymbol{a} \in A$ and observations $\boldsymbol{z} \in Z$ of world state, where $\boldsymbol{a} \in \mathbb{R}^{\ell}$ and $\boldsymbol{z} \in \mathbb{R}^{(m+n)}$. An observation $\boldsymbol{z}$ consists of two components, $\boldsymbol{z} = (\boldsymbol{z}_{\varphi}, \boldsymbol{z}_{\neg\varphi})$, where $\boldsymbol{z}_{\varphi} \in \mathbb{R}^{m}$ describes the robot pose, and $\boldsymbol{z}_{\neg\varphi} \in \mathbb{R}^{n}$ describes any other observables that are of interest to the policy.[1] We define a *demonstration* to consist of a sequence of N_d observations $\{\boldsymbol{z}^j\}_{j=1}^{N_d}$, recorded during teacher execution of the task. A policy $\pi : Z \rightarrow A$ is derived from the collected set

[1] Pose information is necessary for the TPC algorithm, and so $\boldsymbol{z}_{\varphi} \neq \emptyset$. The presence of additional observation information however is application-dependent, and possibly absent such that $\boldsymbol{z}_{\neg\varphi} = \emptyset$.

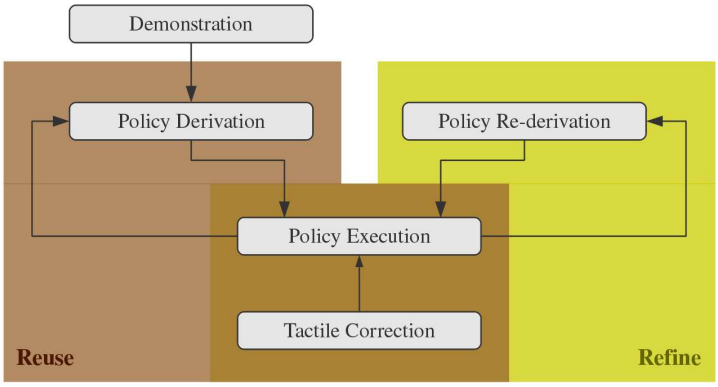

Fig. 2.1 Flow overview of the Tactile Policy Correction algorithm under the operational modes of refinement and reuse.

$D = \{\boldsymbol{z}^j\}_{j=1}^{N}$ of N datapoints from multiple demonstration executions.

2.1 Algorithm Execution

The first phase of the TPC algorithm consists of task demonstration by the teacher, producing dataset D from which the learner derives an initial policy π. The second phase of the algorithm involves learner execution with the policy π, and corrective tactile feedback which is used to adapt π. This *execution-correction-adaptation* cycle continues to the satisfaction of the teacher.

A single execution-correction-adaptation cycle is presented in Algorithm 1. Policy execution (lines 8-10) at timestep t consists of two phases: prediction of a target pose $\hat{\boldsymbol{z}}_{\varphi}^{t}$, and the selection of an action to accomplish that pose. Pose prediction is accomplished via regression techniques, based on state observation $\boldsymbol{z}^{t-1}$ (line 9). Action selection is accomplished via a robot-specific controller, and its execution results in a new robot pose $\boldsymbol{z}_{\varphi}^{t}$ (line 10).

The human teacher may choose to offer a tactile correction at any timestep of the execution. If detected, the robot learner translates the tactile feedback into an incremental shift $\boldsymbol{\delta}_{\epsilon}^{t} \in \mathbb{R}^m$ in robot pose, according to mapping M (line 12). Note that the form taken by the tactile

Algorithm 1 *Tactile Policy Correction*

1: Given D
2: *initialize* $\boldsymbol{\delta}^0 \leftarrow \mathbf{0}$
3: *derive* $\pi \leftarrow$ `policyDerivation`(D)
4: *set* $\boldsymbol{refine} = true \ \vee \ \boldsymbol{reuse} = true$
5:
6: **while** *correcting* **do**
7: *initialize* $\delta_\epsilon^t \leftarrow \mathbf{0}$
8: *Policy* π *execution*:
9: *predict* $\hat{\boldsymbol{z}}_\varphi^t \leftarrow$ `regression`$\left(\boldsymbol{z}^{t-1}\right)$
10: *execute* $\boldsymbol{z}_\varphi^t \leftarrow$ `controller`$\left(\hat{\boldsymbol{z}}_\varphi^t + \boldsymbol{\delta}^{t-1}\right)$
11: **if** $\{$*detect touch*$\}$ **then**
12: *map* $\boldsymbol{\delta}_\epsilon^t \leftarrow M\,(touch)$
13: *correct* $\boldsymbol{z}_\varphi^t \leftarrow$ `controller`$\left(\boldsymbol{z}_\varphi^t + \boldsymbol{\delta}_\epsilon^t\right)$
14: **end if**
15: *record* $\boldsymbol{\delta}^t \leftarrow \boldsymbol{\delta}^{t-1} + \boldsymbol{\delta}_\epsilon^t$
16: **if** $\{\boldsymbol{refine}\}$ **then**
17: *set* w^t
18: *record* $D \leftarrow D \cup \left(\boldsymbol{z}^t, w^t\right)$
19: **else** $\{\boldsymbol{reuse}\}$
20: *select* $D_s \subseteq D$
21: *replace* $\boldsymbol{z}_\varphi^i \leftarrow \boldsymbol{z}_\varphi^i + \boldsymbol{\delta}^t \ , \ \ \forall \boldsymbol{z}^i \in D_s$
22: **end if**
23: **end while**
24:
25: **if** $\{\boldsymbol{refine}\}$ **then**
26: *rederive* $\pi \leftarrow$ `policyDerivation`(D)
27: **return** π
28: **else** $\{\boldsymbol{reuse}\}$
29: *derive* $\pi' \leftarrow$ `policyDerivation`(D)
30: **return** π'
31: **end if**

feedback is platform-specific, depending both on the tactile sensors employed to detect contact and how the sensor feedback is processed.

The robot controller is then passed the new *adjusted* pose, for which the incremental shift $\boldsymbol{\delta}_\epsilon^t$ is added to the current robot pose $\boldsymbol{z}_\varphi^t$ (line 13). The influence of this incremental shift is maintained over multiple timesteps, through an offset parameter $\boldsymbol{\delta}^t \in \mathbb{R}^m$ that maintains a sum of all adjustments seen during the execution (line 15) and is added to the pose prediction at each execution timestep (line 10).

How the pose adjustment is recorded into the policy is handled

differently for policy refinement versus policy reuse:

- For policy *refinement*, the corrected execution is treated as new data for the policy (lines 16-18). Observation $\boldsymbol{z}^t$, and a weight $w^t \in [0, 1]$ for the new datapoint (details in Sec. 2.4.1), are recorded into the set D. The tactile correction thus also is recorded since the current pose, that has responded to the tactile feedback, is recorded through component $\boldsymbol{z}^t_\varphi \in \boldsymbol{z}^t$.

- For policy *reuse*, the indicated correction is applied to existing points within the dataset (lines 19-21). A subset of points $D_s \subseteq D$ are selected, and the offset $\boldsymbol{\delta}^t$ is applied to their pose components $\boldsymbol{z}_\varphi$ (details in Sec. 2.4.2).

In both cases the dataset D is modified and, upon completion of the entire execution, a policy is derived from this set. In the case of refinement, the existing policy π is replaced with an updated version via rederivation (line 26). In the case of reuse, a new policy π' is derived, leaving the original policy π unchanged (line 29).

Important to note is that the TPC algorithm is generic with respect to the techniques used for pose prediction (`regression`) and action selection (`controller`) during policy execution, as well as to the technique that translates tactile feedback into a pose adjustment (mapping M). The following sections (2.2-2.3) will describe the particular techniques we employ for the implementation of the TPC algorithm within this article.

2.2 Policy Execution

This section describes policy execution under our implementation of the TPC algorithm. For pose prediction, Gaussian Mixture Regression is employed (Sec. 2.2.1) along with a modification to allow for execution variability (details in Sec. 2.5), and for action selection an inverse kinematic controller (Sec. 2.2.2).

2.2.1 Pose Prediction

Target poses are predicted through the *GMM-GMR* algorithm (Calinon and Billard, 2007), which first encodes demonstrations in a *Gaussian Mixture Model (GMM)* and then predicts a target pose through *Gaussian Mixture Regression (GMR)* (Cohn et al., 1996). The parameters of the GMM are trained under a weighted version of the *Expectation-Maximization (EM)* algorithm. Full details of the GMM-GMR process and our weighted EM implementation are provided respectively in Tables 2.1 and 2.2.

Our implementation defines observation component $\boldsymbol{z}_{\varphi}$ as the Cartesian position $\boldsymbol{x} \in \mathbb{R}^3$ and orientation $\boldsymbol{q} \in \mathbb{R}^4$ (as a quaternion, $\|\boldsymbol{q}\| = 1$) of the end-effector in a robot-centric reference frame. Thus $\boldsymbol{z}_{\varphi} \equiv [\boldsymbol{x}, \boldsymbol{q}] \in \mathbb{R}^7$. We further define component $\boldsymbol{z}_{\neg\varphi} \equiv \tau \in \mathbb{R}$ as the time of the recorded observation. The GMM thus models the joint probability of the temporal and spatial aspects of the demonstrations. To make a pose prediction, GMR estimates the conditional expectation of $\boldsymbol{z}_{\varphi}$ given $\boldsymbol{z}_{\neg\varphi}$; formally, the expectation $E\left(p\left(\boldsymbol{x}, \boldsymbol{q}|\tau\right)\right)$, also referred to as the marginal joint probability $p_{\tau}\left(\boldsymbol{x}, \boldsymbol{q}\right)$.

The output of GMM-GMR is a mean trajectory and associated covariance envelope. Our formulation additionally takes advantage of the probabilistic nature of this regression technique to generate variability, and thus flexibility, in the predicted trajectory. The details of our approach to deviating from the regression trajectory are provided in Section 2.5.

2.2.2 Action Selection

Given a target pose $\hat{\boldsymbol{z}}_{\varphi}$, action selection is accomplished via an inverse kinematic controller. Our action space A consists of the 7-DoF velocity vector $\dot{\boldsymbol{\theta}} \in \mathbb{R}^7$ controlling the joint angles of a robot arm. The manipulator of our implementation (Sec. 3.1) is redundant, as the number of degrees of freedom (7) exceeds the number of constraints (6, position and orientation). We therefore compute desired joint angle velocities $\dot{\boldsymbol{\theta}}$ according to the distance between the target pose $\hat{\boldsymbol{z}}_{\varphi}^{t}$ and the current robot pose $\boldsymbol{z}_{\varphi}^{t}$ by using a pseudo-inverse method that both avoids joint limits and is robust to singularities (Baerlocher and Boulic, 2004).

Table 2.1 Gaussian Mixture Regression (GMR)

The demonstrations within dataset D are modeled probabilistically within a *Gaussian Mixture Model* (GMM), that defines for each point $\boldsymbol{z}^j \in D$ a probability function given by a mixture of K Gaussian components

$$p(\boldsymbol{z}^j) \;=\; \sum_{k=1}^{K} p(k)\, p(\boldsymbol{z}^j|k) \;=\; \sum_{k=1}^{K} \gamma_k\, \mathcal{N}\left(\boldsymbol{z}^j; \mu_k, \Sigma_k\right)$$

where γ_k is the prior of the k^{th} component, and $\mathcal{N}\left(\boldsymbol{z}^j; \mu_k, \Sigma_k\right)$ is a Gaussian distribution with mean μ_k and covariance Σ_k. The optimal number of components K is determined according to the *Bayes Information Criterion (BIC)*. In this work, a datapoint $\boldsymbol{z}^j$ consists of two parts: pose $\boldsymbol{z}^j_\varphi$ ($\equiv [\boldsymbol{x}, \boldsymbol{q}]$) and timestamp $\boldsymbol{z}^j_{\neg\varphi}$ ($\equiv \tau$), which for notational simplicity we reference here respectively with φ and τ. To compute a conditional expectation of φ given τ (i.e. $\boldsymbol{z}_\varphi$ given $\boldsymbol{z}_{\neg\varphi}$) for each component k, we first define

$$\mu_k \;=\; \left(\, \mu_{\tau,k},\; \mu_{\varphi,k} \,\right), \qquad \Sigma_k \;=\; \begin{pmatrix} \Sigma_{\tau\tau,k} & \Sigma_{\tau\varphi,k} \\ \Sigma_{\varphi\tau,k} & \Sigma_{\varphi\varphi,k} \end{pmatrix}$$

and *Gaussian Mixture Regression (GMR)* then uses

$$p\left(\varphi_k|\tau, k\right) \;\sim\; \mathcal{N}\left(\varphi_k; \hat{\mu}_{\varphi,k}, \hat{\Sigma}_{\varphi\varphi,k}\right)$$

$$\begin{aligned} \hat{\mu}_{\varphi,k} &= \mu_{\varphi,k} + \Sigma_{\varphi\tau,k}\left(\Sigma_{\tau\tau,k}\right)^{-1}\left(\tau - \mu_{\tau,k}\right) \\ \hat{\Sigma}_{\varphi\varphi,k} &= \Sigma_{\varphi\varphi,k} - \Sigma_{\varphi\tau,k}\left(\Sigma_{\tau\tau,k}\right)^{-1}\Sigma_{\tau\varphi,k} \end{aligned}$$

to provide the expected distribution of φ given τ for component k (i.e. $\boldsymbol{z}_{\varphi,k}$ given $\boldsymbol{z}_{\neg\varphi}$) . Finally, by considering all of the components k and their regression priors $\beta_k(\tau)$, a target pose $\hat{\boldsymbol{z}}_\varphi$ is predicted with mean $\hat{\mu}_\varphi$ and covariance $\hat{\Sigma}_{\varphi\varphi}$ according to

$$\hat{\mu}_\varphi = \sum_{k=1}^{K} \beta_k\left(\tau\right)\hat{\mu}_k \quad , \quad \hat{\Sigma}_{\varphi\varphi} = \sum_{k=1}^{K} \beta_k\left(\tau\right)^2 \hat{\Sigma}_{\varphi\varphi,k}$$

$$\beta_k\left(\tau\right) \;=\; \frac{p\left(k\right)p\left(\tau|k\right)}{\sum_{i=1}^{K} p\left(i\right)p\left(\tau|i\right)} \;=\; \frac{\gamma_k\,\mathcal{N}\left(\tau; \mu_{\tau,k}, \Sigma_{\tau\tau,k}\right)}{\sum_{i=1}^{K}\mathcal{N}\left(\tau; \mu_{\tau,i}, \Sigma_{\tau\tau,i}\right)}$$

2.3 Tactile Corrections

Our interface for providing tactile corrections to the robot learner consists of five *Ergonomic Touchpads* located on the manipulator arm.[2]

[2] Touchpad feedback is somewhat limited in comparison to more sophisticated tactile sensors. In practice corrective repositioning is not always as responsive as the teacher requires, and so we pause policy execution such that psuedo-code lines 12-13 loop until repositioning is complete. Note that this limitation results from a deficiency in hardware, not the algorithm. The validation of TPC with a more sophisticated tactile sensor is under active

Table 2.2 Weighted Expectation-Maximization (EM)

Our weighted version of the EM algorithm modifies GMM-GMR parameter estimation to include weight $w^j \geq 0,\ \Sigma_{j=1}^{N} w^j > 0$. The algorithm loops between the *E-step* and *M-step* until the overall likelihood $\sum_{k=1}^{K} E_k$ is maximized:
E-step:

$$\begin{aligned} p_{k,j}^{(i+1)} &= \frac{\gamma_k^{(i)} \mathcal{N}\left(\boldsymbol{z}^j; \mu_k^{(i)}, \Sigma_k^{(i)}\right)}{\Sigma_{i_k=1}^{K} \gamma_{i_k}^{(i)} \mathcal{N}\left(\boldsymbol{z}^j; \mu_{i_k}^{(i)}, \Sigma_{i_k}^{(i)}\right)} \\ E_k^{(i+1)} &= \Sigma_{j=1}^{N} w^j p_{k,j}^{(i+1)} \end{aligned}$$

M-step:

$$\begin{aligned} \gamma_k^{(i+1)} &= \frac{E_k^{(i+1)}}{\Sigma_{j=1}^{N} w^j} \\ \mu_k^{(i+1)} &= \frac{\Sigma_{j=1}^{N} w^j p_{k,j}^{(i+1)} \boldsymbol{z}^j}{E_k^{(i+1)}} \\ \Sigma_k^{(i+1)} &= \frac{\Sigma_{j=1}^{N} w^j p_{k,j}^{(i+1)} \left(\boldsymbol{z}^j - \mu_k^{(i+1)}\right)\left(\boldsymbol{z}^j - \mu_k^{(i+1)}\right)^T}{E_k^{(i+1)}} \end{aligned}$$

The pads detect contact presence and relative motion, which we map to a change in end-effector position and orientation.

Four touchpads, $T_0 \cdots T_3$, encircle the lower forearm of the robot arm (near the wrist), and one, T_4, is located on the back of the robot hand (Fig. 2.2a,b). Touch data from pad $T_k, k = 0..4$, consists of a 2-D relative change in pixels $\left(\Delta u_k^t, \Delta v_k^t\right)$. The target pose adjustement $\boldsymbol{\delta}_\epsilon^t$ is computed using the forward kinematic function f of the whole arm, such that $\boldsymbol{\delta}_\epsilon^t = f(\boldsymbol{\theta}^t + \dot{\boldsymbol{\theta}}^t \Delta t) - \hat{\boldsymbol{z}}_\varphi^t$. Here $\boldsymbol{\theta}^t$ is the current joint configuration, Δt the timestep for touchpad data capture, $\hat{\boldsymbol{z}}_\varphi^t$ the target pose predicted by the regression model, and $\dot{\boldsymbol{\theta}}^t$ the joint velocity to accomplish the adjustment, the computation for which is described next. In practice, we decompose the mapping $M \mapsto \boldsymbol{\delta}_\epsilon$ into two distinct parts that operate separately on the wrist and hand, as this seemed a more intuitive mapping for the experimenters providing corrections.

The first part of the mapping M operates on the first 5-DoF leading to the wrist of our 7-DoF manipulator. Sliding the fingers along two

development for future work.

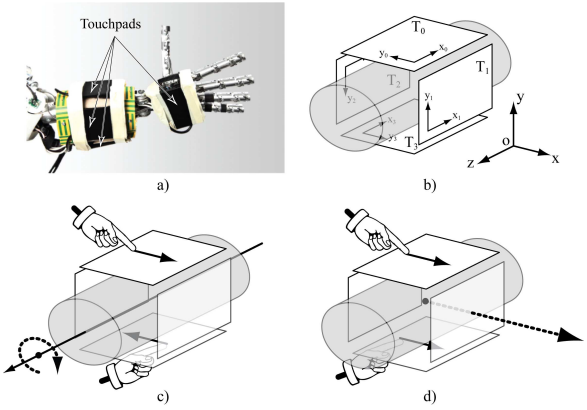

Fig. 2.2 a,b) Schematic of the touch pads controlling the robot wrist and hand. c,d) Fingers sliding on opposite pads produces rotational (c) or translational (d) motions.

opposite touchpads leads either to a translational or rotational motion command, depending on whether the sliding directions agree or not (Fig. 2.2c,d). The velocity $\dot{\boldsymbol{z}}^t_\varphi$ for the pose correction is computed by mapping touch data (in $\mathbb{R}^8$, 4 pads $\times$ 2-D data) from pads $T_0 \cdots T_3$ to a vector describing the target velocity in Cartesian-space wrist coordinates, and then to robot-centric world coordinates through rotation matrix R:

$$\dot{\boldsymbol{z}}^t_\varphi \;=\; \begin{bmatrix} R \\ R \end{bmatrix} \begin{bmatrix} \kappa_\nu \left(-\Delta v^t_0 + \Delta v^t_2\right) \\ \kappa_\nu \left(\Delta v^t_1 - \Delta v^t_3\right) \\ \kappa_\nu \left(-\Delta u^t_0 - \Delta u^t_1 - \Delta u^t_2 - \Delta u^t_3\right) \\ \kappa_\omega \left(-\Delta u^t_0 + \Delta u^t_2\right) \\ \kappa_\omega \left(\Delta u^t_1 - \Delta u^t_3\right) \\ \kappa_\omega \left(\Delta v^t_0 + \Delta v^t_1 + \Delta v^t_2 + \Delta v^t_3\right) \end{bmatrix}$$

Constant parameters κ_ν and κ_ω scale respectively the translational and rotational components of the touch data, to account for differences in units (pixels for the tactile feedback, $\frac{m}{s}$ and $\frac{rad}{s}$ for the velocity components). The mapping from Cartesian-space velocity $\dot{\boldsymbol{z}}^t_\varphi$ to joint velocity $\dot{\boldsymbol{\theta}}^t_{\{0..5\}}$ for the first 5-DoFs in the arm then is computed using inverse kinematics (Baerlocher and Boulic, 2004).

The second part of the mapping M operates on the last 2-DoF of the manipulator, that control the robot hand. Touch data (in $\mathbb{R}^2$, 1

pad × 2-D data) from pad T_4 thus maps directly to the target joint velocities, such that $\dot{\boldsymbol{\theta}}_{\{6..7\}} = \left[\kappa_\nu \Delta u_4^t, \kappa_\nu \Delta v_4^t\right]$.

2.4 Policy Adaptation

Upon the completion of a corrected execution, policy adaptation is accomplished by (re)deriving the policy from the feedback-modified dataset. How the dataset has been modified depends on whether the policy is being adapted for the purpose of refinement (Sec. 2.4.1) or reuse (Sec. 2.4.2). The operational mode for the algorithm, being either refinement or reuse, is indicated by the teacher (Alg. 1, line 4).

Policy (re)derivation consists of (re)estimating the regression parameters, again using the weighted EM algorithm (Tbl. 2.2). Though policy execution under TPC consists both of pose prediction via regression techniques and action selection by a controller, under our implementation the controller is statically defined. Policy derivation therefore requires regression parameter estimation only.

2.4.1 Adaptation for Policy Refinement

When tactile corrections are provided for the purpose of policy refinement, new datapoints are generated by the execution-correction process. A weight is associated with each point in the set D, and therefore must be determined for any new datapoints as well.

Datapoint weights are assigned based on the covariance envelope of the original GMM derived from the demonstration data. In particular, we define weight functions for corrected executions $w_C(t)$ and demonstrated executions $w_D(t)$ as

$$w_C(t) = 1 - \frac{|\hat{\Sigma}^t_{\varphi\varphi}|^{\frac{1}{2}}}{2 \cdot \Sigma_{max}} \quad , \quad \Sigma_{max} = \max_t |\hat{\Sigma}^t_{\varphi\varphi}|^{\frac{1}{2}} \tag{2.1}$$

$$w_D(t) = 1 - w_C(t) \tag{2.2}$$

where $|\hat{\Sigma}^t_{\varphi\varphi}|$ is the determinant of the GMR prediction covariance matrix at time t. We then assign weight w^j for datapoint $\boldsymbol{z}^j$ with functions $w_D(t)$ or $w_C(t)$, based on whether $\boldsymbol{z}^j$ was part of a demonstrated or corrected execution (respectively) and the time ($\tau \equiv \boldsymbol{z}^j_{\neg\varphi}$) of the observation recording.

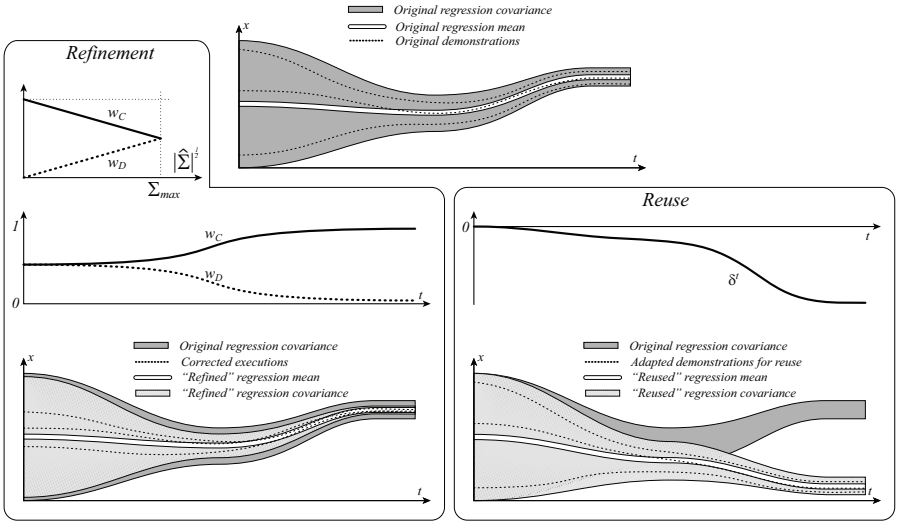

Fig. 2.3 Illustration of policy adaptation under refinement and reuse. Top center: Original demonstration data, with associated regression mean and covariance envelope. Refinement panel: Our weight function formulation (top), that is a function of covariance envelope size ($|\hat{\Sigma}|^{\frac{1}{2}}$). Illustration of an example weight function (middle) and how with it the covariance envelope narrows more dramatically as time progresses and $w_C \gg w_D$ (bottom). Reuse panel: Illustration of the accumulation of correction offsets during an execution (top), and how this shifts the points in the dataset and thus the regression signal (bottom).

With this weight formulation, we assume teacher demonstrations provide an accurate portrayal of the variability profile of the task. That is, in areas of low covariance, little variability is allowed (or equivalently, high precision is required) in the target task behavior, while in areas of high covariance, much variability in the resulting behavior is acceptable, even expected. With our weight formulation, in areas of low covariance ($|\hat{\Sigma}^t_{\varphi\varphi}|^{\frac{1}{2}} \ll \Sigma_{\max}$), corrected datapoints are given a high weight, and the regression signal accordingly shifts strongly. By contrast, in areas of high covariance ($|\hat{\Sigma}^t_{\varphi\varphi}|^{\frac{1}{2}} \to \Sigma_{\max}$), it is not unexpected that executions might differ from the demonstrated behavior, and so demonstrated and corrected execution points are given approximately equal weight. In Figure 2.3 (*Refinement* panel) this weight formulation is shown (top), as well as an example weight function (middle) and the resulting adapted regression signal (bottom).

2.4.2 Adaptation for Policy Reuse

When tactile corrections are provided for the purpose of policy reuse, existing points within the set D are modified. In particular, a subset of points $D_s \subseteq D$ are selected, and the pose adjustment is applied to the pose component of these points ($\boldsymbol{z}_\varphi \in D_s$). Note that an entirely new policy is instantiated when reuse is employed, and it is not expected after reuse that the new policy be able to perform the task of the original policy from which it was adapted.

The subset D_s is selected according to nearness, within the input space of the regression function, between the execution point that received the tactile correction and the points within dataset D. In our work the input space of the regression function is execution time. Since our demonstrations are resampled to have an equal number of execution points, the metric for *nearness* is straightforward: for a given modified execution point $\boldsymbol{z}^t$, we build D_s by simply taking all points in D that occurred at this same time in their respective demonstration trajectories, such that $D_s = \{\boldsymbol{z}^i | \boldsymbol{z}^i_{\neg\varphi} = \boldsymbol{z}^t_{\neg\varphi}, \forall \boldsymbol{z}^i \in D\}$. We then apply to the pose components of these points the offset $\boldsymbol{\delta}^t$.

With this nearness metric however, caution must be exercised when changing points within the dataset. In particular, our regression formulation (details in the following section, 2.5) allows for deviations from srictly following the regression signal - that is, the mean trajectory - of GMR. Thus at the same point in time with respect to the execution sequence, different executions might be in distinct areas of the state space for which the target policy behavior differs. Caution must be exercised since a correction which is appropriate for one location might not be appropriate for the other. Consider for example a policy for object grasping, where at the time just prior to grasping a different hand orientation is required depending on the direction of approach. A correction that flips the robot hand by 90 degrees thus might be appropriate if the object was approached from the top (causing the object to be grasped from the side), but not if approached from the side (causing an attempted grasp from below, and a collision with the object's supporting surface).

We address this issue by restricting the operational mode of reuse to

correcting only executions that follow exactly the regression signal. This restriction ensures that an execution point receiving corrections lies at the regression mean of the set of datapoints with similar timestamps. Any provided corrections thus produce offsets that are appropriate for this mean, and are not particular to the extremes of these points. Note that this restriction is in place only for reuse, and is lifted for executions that are a straightforward reproduction or corrected for the purpose of refinement.

In conclusion, the idea behind the TPC formulation for reuse is to take one large step in the direction of the new policy behavior, by shifting entire subsets of the existing dataset. By comparison, if the modified execution was instead added to the existing dataset, as in refinement, the new data would simply be averaged with the existing data during policy derivation. While the regression trajectory would indeed be pulled in the direction of the new data, and thus the new target behavior, the effect would be more iterative and less dramatic than one-shot reuse. In Figure 2.3 (*Reuse* panel) an illustration is provided of the correction offsets accumulated throughout an execution (top), and the resulting shift in regression signal (bottom).

2.5 Deviating from the Regression Signal

We conclude this chapter with a description of our formulation that allows for flexibility in the trajectory predicted by GMM-GMR.

2.5.1 Formulation

Under GMR, a target pose $\hat{\boldsymbol{z}}_{\varphi}^{t}$ is predicted with mean $\hat{\boldsymbol{\mu}}_{\varphi}^{t}$ and covariance $\hat{\boldsymbol{\Sigma}}_{\varphi\varphi}^{t}$ (Fig. 2.4). We modify the pose prediction by

$$\hat{\boldsymbol{z}}_{\varphi}^{t} = \hat{\boldsymbol{\mu}}_{\varphi}^{t} + \boldsymbol{\delta}_{\lambda}^{t} \tag{2.3}$$

and thus apply to the regression mean offset $\boldsymbol{\delta}_{\lambda}^{t} \in \mathbb{R}^{m}$

$$\boldsymbol{\delta}_{\lambda}^{t} = \begin{cases} \Delta_{\lambda}^{t} & if \quad \lambda^{t} \leq \lambda_{\max} \\ \Delta_{\lambda}^{t} \frac{\lambda_{\max}}{\lambda^{t}} & otherwise \end{cases} \tag{2.4}$$

$$\Delta_{\lambda}^{t} = \boldsymbol{z}_{\varphi}^{t} - \hat{\boldsymbol{\mu}}_{\varphi}^{t} \quad , \quad \lambda^{t} = \|(\hat{\boldsymbol{\Sigma}}_{\varphi\varphi}^{t})^{-\frac{1}{2}} \Delta_{\lambda}^{t}\| \tag{2.5}$$

where $\boldsymbol{\delta}_{\lambda}^{t}$ is defined by the difference (Δ_{λ}^{t}) between the current robot pose and regression mean, and whether the magnitude (λ^{t}) of this difference (inversely scaled by standard deviation ($\hat{\boldsymbol{\Sigma}}_{\varphi\varphi}^{t}$)$^{\frac{1}{2}}$) exceeds a threshold ($\lambda_{\max}$).

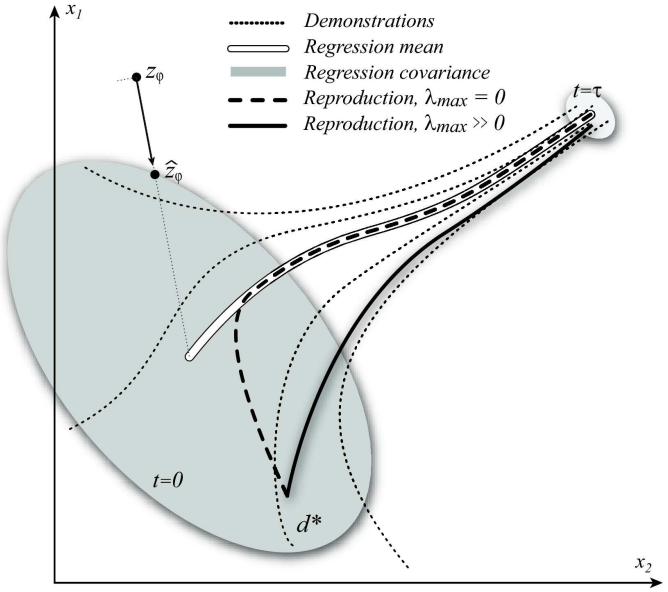

Fig. 2.4 Illustration of our offset formulation for GMR that allows for deviations from the regression mean (bold *vs*. dashed lines), showing adaptability with respect starting position.

The amount of allowable deviation is dictated in terms of an acceptable number ($\lambda_{\max}$) of standard deviations from the regression mean, where $\lambda_{\max} \geq 0$ is a constant parameter set by hand (in our empirical validations, $\lambda_{\max} = 2$). For execution points (including starting positions) within this threshold (i.e. within $\lambda_{\max}$ standard deviations of the regression mean $\hat{\boldsymbol{\mu}}_{\varphi}^{t}$), the execution proceeds with its current pose (i.e. $\hat{\boldsymbol{z}}_{\varphi}^{t} = \boldsymbol{\mu}_{\varphi}^{t} + \Delta_{\lambda}^{t} = \boldsymbol{z}_{\varphi}^{t}$). Execution points outside of this threshold are first projected (e.g. Fig. 2.4, $\boldsymbol{z}_{\varphi}$ to $\hat{\boldsymbol{z}}_{\varphi}$) to the envelope (shaded region) defined by $\lambda_{\max}$ standard deviations around the regression mean. The result is more flexible learner executions, that take advantage of the variability present within the teacher demonstrations.

2.5.2 More Flexible Executions

One gain of this regression formulation is allowing the learner execution to take a more direct path to the goal, that perhaps deviates from the regression trajectory but is still within the bounds of what was demonstrated. Figure 2.4 illustrates that in the absence of offset $\boldsymbol{\delta}_\lambda$, the execution trajectory (dashed line) follows the regression mean (white trajectory), regardless of whether a more appropriate path (e.g. a shorter path, such as demonstration d^*) is contained within the set of demonstrations. With the offset, however, the learner execution is free to follow a more direct path to the goal (bold line), providing this is within $\lambda_{\max}$ standard deviations of the regression mean.

The executions in Figure 2.5 confirm this behavior with real robot data. Here the validation task consisted of positioning the 7-DoF end-effector of the iCub humanoid robot to grasp a cylindrical object.[3] Demonstrations were provided from multiple starting end-effector positions with respect to the object. To explore policy *flexibility* with respect to acceptable variability in task execution, three policies were developed for comparison:

π : Derived from the demonstration set using standard GMR.

π_λ : Derived from the demonstration set using our modified version of GMR with offset $\boldsymbol{\delta}_\lambda$.

$\pi_{\lambda,c}$: Produced from the tactile correction of π_λ using TPC.

Table 2.3 provides the lengths of the execution trajectories (as fractions of the distance traveled by policy π) from 4 starting positions ($s_1..s_4$) for all policies. Indeed, from all positions the incorporation of offset $\boldsymbol{\delta}_\lambda$ allows for execution paths that approach the target position more directly, shown by shorter trajectory lengths (π_λ vs. π, $\pi_{\lambda,c}$ vs. π). The most dramatic improvement is seen with starting point s_4, whose position is such that the execution must travel explicitly away from the target position ($*$) to reach the start of the regression trajectory (s_r). In this case overt backtracking is the result if offset $\boldsymbol{\delta}_\lambda$ is not employed.

[3] Full details of the iCub robot and this experimental domain will be provided in Section 3.1.

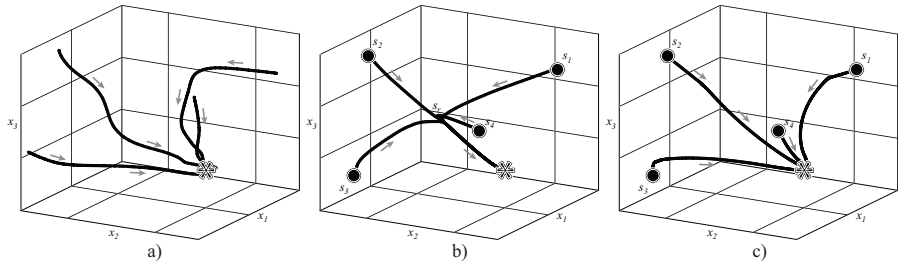

Fig. 2.5 a) Demonstration executions to target position $*$. b) Executions from starting positions $s_1..s_4$, performed by policy π. Note that executions first visit the start (s_r) of the regression trajectory. c) Executions from starting positions $s_1..s_4$, performed by policy π_λ, which proceed directly to the target position.

Unnecessary backtracking in the absence of $\boldsymbol{\delta}_\lambda$ is a consequence of time-dependence in the system. With our offset, the pose predictions are no longer restricted to follow exactly the regression trajectory, but are still constrained by the demonstrations in the set. Namely, if the starting position of the current execution is outside of the initial covariance envelope, and thus sufficiently dissimilar to any of the demonstration start positions, the execution will first snap (possibly backtracking) to the closest point on the edge of this initial envelope; by contrast, without offset λ the execution would snap all the way to the regression mean. The offset $\boldsymbol{\delta}_\lambda$ formulation therefore tackles to a certain degree some of the negative consequences of time-dependence, though time-dependence is still present and at times a drawback.

Table 2.3 Execution Length (from multiple starting positions, as a fraction of the execution length of policy π)

Starting Position	π	π_λ	$\pi_{\lambda,c}$
s_1	1	0.69	**0.66**
s_2	1	**0.88**	**0.88**
s_3	1	**0.64**	0.67
s_4	1	0.35	**0.27**

3

Empirical Validation

This chapter provides empirical validation of the TPC algorithm. Our experimental domain involves positioning the end-effectors of a high-DoF humanoid, for interactions with and between a variety of objects. The performance of policies refined under TPC is reported, and successful policy reuse also is confirmed. We furthermore examine shifts in the regression covariance envelope, which as a result of tactile feedback may contract or expand within different dimensions to increase respectively execution precision or flexibility. A comparison additionally is provided between policies developed under TPC, and those that receive more teleoperation demonstrations in lieu of tactile corrections.

We have implemented the TPC algorithm on a small 53-DoF humanoid, the iCub (Tsagarakis et al., 2007). Demonstration is performed via teleoperation by a human teacher, which is non-trivial as simultaneous control of 7 degrees of freedom is required to teleoperate a single arm, 14 to teleoperate both arms simultaneously. Teleoperation is accomplished through a joint recording system and a mapping that allows the human to directly control the motion of the robot arm by moving his own arm, during which the robot records from its own sensors (Fig. 3.1). Sensing units from the commercial $XSens$ joint recording

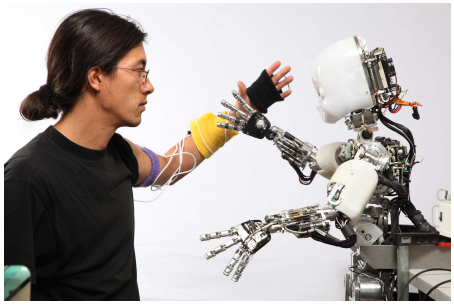

Fig. 3.1 Teleoperation of the iCub robot by mapping the human joint angles, and thus the human arm movement, to the robot arm.

system are placed on the human's upper and lower arm, and back of the hand. Each unit contains an accelerometer, gyroscope and inertial sensing unit, and provides orientation information that we translate into human joint angles. We then map the human joint angles to the joint angles of the robot arm, thus accomplishing remote control.

In each of the following experiments, policy development consists initially of task demonstration, followed by tactile corrections. Two human teachers provide demonstrations and corrections, neither of whom are robotics novices.

3.1 Experimental Setup 1: Grasp Positioning

For our first set of validation tasks, the robot learns to position the end-effector(s) of its 7-DoF arm(s) for uni-manual and bi-manual grasping of different objects. Closing the hand(s) for grasping is handled by a static controller.[1] Multiple policies are developed to accomplish various end-effector positioning behaviors, each of which has the learner position one or both of its end-effectors to grasp an object located at a particular position within the robot-centric coordinate frame.[2]

[1] The focus of the task objective is on end-effector positioning, rather than the grasp itself, since the iCub hand has no force sensors or tactile feedback in its hands. Note also that if controlling the hand is a part of the demonstrations, then the joint space is 15-DoF for each arm and a more complex teleoperation system is required.

[2] The location of an object is fixed with respect to the robot for each developed policy. This construction easily extends to be flexible with respect to object position however, by switching to an object-oriented coordinate frame. As our goal was to validate policy

3.1.1 Evaluation of Policy Performance

The performance of a policy is evaluated according to whether the end-effector is positioned such that the robot is able to grasp the object. Positions within the final covariance envelope of each policy are tested for success in grasping, and are selected both systematically and randomly; in particular, positions are systematically selected along the boundaries of the covariance envelope, and sampled randomly within its bounds. The same set of positions S are employed across all policies, scaled by the respective dimensions of the covariance envelope for each.

In particular, the set S contains the following 21 positions: the final position of the regression trajectory (1), the extremums of the final covariance envelope (14, 2 extremums $\times$ 7 pose dimensions) and random positions within the covariance envelope (6). The extremum positions are determined by setting a single dimension to its largest and smallest values within the covariance envelope, and setting all other dimensions to their regression mean values (i.e. the regression mean $\pm$ the covariance value of the dimension under consideration). We examine these extremum positions by looking separately at the subset of positions corresponding to end-effector position ($S_p \subset S, |S_p| = 6$) and end-effector orientation ($S_o \subset S, |S_o| = 8$). The reason for taking particular interest in performance at the covariance envelope boundaries arises from our flexible regression formulation: with offset $\boldsymbol{\delta}_\lambda$, the regression signal is not restricted to follow only the regression mean, and produces predictions within or at the boundaries of the covariance envelope. Furthermore, the performance within the envelope (i.e. on the mean and random positions), tends to be quite good and vary little across policies.

3.1.2 Analysis of the Covariance Envelope

When the TPC algorithm is in refinement mode, tactile corrections produce new data, which might constrict or expand the covariance envelope of the regression signal. When the envelope is constricted, the

refinement and reuse under TPC, we chose a simpler task representation that was not complicated by the sensing requirements to detect object position.

resulting motion of the robot becomes more constrained and, assuming a good regression trajectory, thus also more precise. When the envelope is expanded, the motion becomes less constrained and thus more flexible. Either might be desirable or undesirable within different dimensions for a given task.

We will examine changes in covariance by looking at the normalized standard deviation of the full covariance matrix, as well as the submatrices corresponding to end-effector position and orientation. More specifically, the full covariance $\hat{\Sigma}$ is composed of four submatrices

$$\hat{\Sigma} = \begin{bmatrix} \hat{\Sigma}_x & \hat{\Sigma}_{qx} \\ \hat{\Sigma}_{xq} & \hat{\Sigma}_q \end{bmatrix} \tag{3.1}$$

where subscripts x and q refer respectively to the position and orientation components of the robot pose. The normalized covariance is computed as $|\Sigma|^{\frac{1}{2N}}$ (where $|\Sigma|$ is the determinant of the $N \times N$ matrix Σ), and is reported for a given full covariance matrix $\hat{\Sigma}$ ($N = 7$) and its position and orientation submatrices $\hat{\Sigma}_x, \hat{\Sigma}_q$ ($N = 3, 4$).

The position dimensions of the covariance envelope will be examined in further detail, by looking at the change in envelope shape at the end of the motion trajectory. In particular, we consider to what extent the envelope shape deviates from a sphere, which corresponds to equal variability in all three position dimensions. We measure this deviation according *ellipsoid level*, defined as $\lambda_1/\sqrt{\lambda_2\lambda_3}$ where $\lambda_1^2 \geq \lambda_2^2 \geq \lambda_3^2$ are the eigenvalues of $\hat{\Sigma}_x$. Intuitively, this metric compares the length of the ellipsoid's longest axis (λ_1) to the bounding box (more specifically, the square root[3] of the area of the bounding box, $\sqrt{\lambda_2\lambda_3}$) of the cross section perpendicular to this axis.

3.2 Refinement

We begin with an examination of policy refinement. Policies for four end-effector positioning behaviors are developed by first demonstrating the behavior, and then providing tactile corrections.

[3] The square root corrects for comparing a length (λ_1) to an area ($\lambda_2 \cdot \lambda_3$).

3.2.1 Policy Development and Evaluation

We examine the effects of refinement by contrasting a policy before and after tactile correction, and comparing it also to a policy developed using teleoperation demonstration exclusively. The object behaviors considered include positioning (e.g. Fig. 3.2) the right end-effector for grasping a ball (π_b) and a cylinder (π_c), and positioning both the left and right end-effectors for grasping a tray (π_r, π_l).[4] For each object behavior, three policies are derived (Tbl. 3.1): the first from a set of 4 demonstrations, the second from that demonstration set plus tactile corrections, and the third from that demonstration set plus 4 additional demonstrations.

Table 3.1 Notational summary for the policies developed to evaluate refinement.

	Ball	Cylinder	Tray, right	Tray, left
4 Demos	π_b^{4d}	π_c^{4d}	π_r^{4d}	π_l^{4d}
4 Demos + Refine	$\pi_b^{4d'}$	$\pi_c^{4d'}$	$\pi_r^{4d'}$	$\pi_l^{4d'}$
4 Demos + 4 Demos	π_b^{8d}	π_c^{8d}	π_r^{8d}	π_l^{8d}

3.2.2 Performance Improvement

The performance of all policies was found to improve following tactile refinement (Tbl. 3.2, π_i^{4d} vs. $\pi_i^{4d'}, i = \{b, c, l, r\}$). Averaged over all policy behaviors, performance improved from a success rate of $81.0 \pm 8.7\%$ for the policies derived from 4 demonstrations, to $92.9 \pm 6.2\%$ after those policies were provided with tactile corrections.

Tactile refinement furthermore was found to be more effective at improving policy performance than providing more teleoperation demonstrations (Tbl. 3.2, $\pi_i^{4d'}$ vs. $\pi_i^{8d}, i = \{b, c, l, r\}$). While performance on average improved following tactile refinement, by contrast it *declined*

[4] Note that the demonstrations of tray grasping are performed separately for the right and left arms. While simultaneous operation is feasible technically with our teleoperation system, it is difficult for the teacher to control both arms simultaneously and as a consequence demonstration quality is lower than it is with separate demonstrations.

Table 3.2 Performance results, comparing tactile refinement to more teleoperation.

	Mean (%)	Extremums (%)		Random (%)	Full set S (%)
		Position S_p	Orientation S_o		
π_b^{4d}	100	33.3	75	100	71.4
$\pi_b^{4d'}$	100	83.3	100	100	95.2
π_b^{8d}	100	33.3	75	83.33	66.7
π_c^{4d}	100	50	100	100	85.7
$\pi_c^{4d'}$	100	100	100	100	100
π_c^{8d}	100	50	87.5	66.67	71.4
π_{tr}^{4d}	100	50	75	100	76.2
$\pi_{tr}^{4d'}$	100	66.7	87.5	100	85.7
π_{tr}^{8d}	100	50	87.5	83.33	76.2
π_{tl}^{4d}	100	83.3	87.5	100	90.5
$\pi_{tl}^{4d'}$	100	83.3	87.5	100	90.5
π_{tl}^{8d}	100	66.7	87.5	100	85.7

with more teleoperation demonstrations, from 81.0±8.7% to 75.0±8.1% (average over all policy behaviors). The likely cause is growth in covariance (discussed in the following section) which, paired with the decrease in performance, implies that these demonstrations introduced *undesirable* variability into the dataset. In general, providing more demonstrations with our teleoperation system increased the covariance envelope, as very precise executions were difficult to achieve. When the learner has limited information about the task behavior in many areas of the execution space, providing more demonstrations typically resulted in an increase in policy performance, despite the growth in covariance. However, once the policy was sufficiently informed, especially in areas where precise positioning was required, then the lack of precision in the teleoperation interface, as well as the noise in human execution, was more likely to introduce unwanted variability into the policy. Changes in the covariance envelope, and its effect on policy performance, are discussed next.

3.2.3 Adapting the Covariance Envelope

Table 3.3 compares the changes in covariance following refinement versus more teleoperation demonstration, by reporting the normalized

standard deviations matrices. In particular, tactile refinement reduced the standard deviation of the regression signal ($\pi^{4d'}$ vs. π^{4d}), where by contrast providing more demonstrations consistently increased the standard deviation (π^{8d} vs. π^{4d}). Given that tactile refinement also improved policy performance, while more demonstrations negatively impacted performance (Tbl. 3.2), we conclude that refinement *removed*, while more demonstration *introduced*, unwanted variability into the policy behavior.

Variability with respect to the starting position was present in the original demonstration sets. The cylinder and tray tasks however also allowed for some variability in the *target* position, as the hand may be positioned for grasping at various locations along the principle axis of the cylinder or edge of the tray. Variability in target position was minimally present in the demonstration set, since navigating the end-effector to various grasp locations on the cylinder required a high level of precision that was difficult to achieve with the mechanism used for teleoperation. Through tactile corrections, however, the teacher *was* able to convey variability with respect to target position.

Stated more generally, it can be the case that in areas requiring high precision (e.g. at the target position) a broadened covariance is desirable along certain dimensions (e.g. along the length of the cylinder), while a narrowed covariance is desirable along others (e.g. location of the cylinder). Our teleoperation system was unable to isolate its operation to a single dimension in such high-precision areas, and so broadened the covariance within all dimensions. By contrast, the tactile correction interface was sensitive enough to operate within a single dimension in high-precision areas, and so broadened the covari-

Table 3.3 Normalized standard deviation, average over all policy behaviors. The full covariance over all dimensions is shown, as well as the covariance over those corresponding to position only and orientation only.

	Standard Deviation ($\times 10^{-2}$)		
	Full $\hat{\Sigma}$	Position only $\hat{\Sigma}_x$	Orientation only $\hat{\Sigma}_q$
π^{4d}	1.4 ± 0.3	1.0 ± 0.3	2.5 ± 0.8
$\pi^{4d'}$	1.0 ± 0.4	0.9 ± 0.2	1.5 ± 0.7
π^{8d}	2.0 ± 0.1	1.5 ± 0.2	3.3 ± 0.7

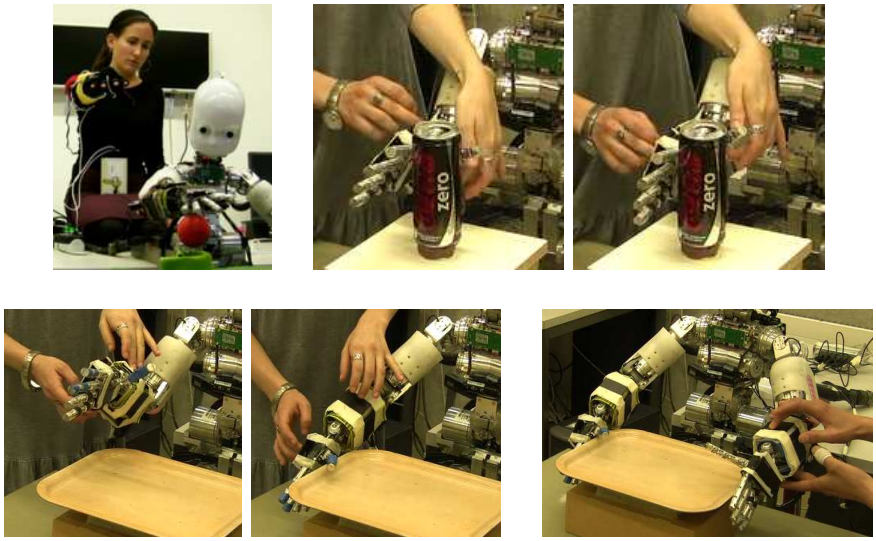

Fig. 3.2 Sequence of tasks learned from policy reuse. Left to right, top to bottom: Demonstration of ball-grasping via teleoperation (π_b); reuse of ball-grasping to grasp a cylinder (π_c); reuse of cylinder-grasping to grasp a tray with the right hand (π_r); mirroring of right-handed tray-grasping to grasp a tray with the left hand (π_l).

ance within only select dimensions.

An increase in flexibility within a single dimension is reflected in an increase in ellipsoid level. This was seen following tactile corrections (3.0 ± 1.1 vs. 2.1 ± 0.4, for policies $\pi_i^{4d'}$ vs. π_i^{4d}, average over $i = \{c, r, l\}$). Providing more teleoperation demonstrations however was not able to increase the ellipsoid level (1.9 ± 0.5 vs. 2.1 ± 0.4, for policies π_i^{8d} vs. π_i^{4d}, average over $i = \{c, r, l\}$), though the teacher was in fact making an effort to indicate flexibility when appropriate.

3.3 Reuse: Efficient Sequence

We next examine policy reuse, by learning policies for the four object behaviors of the previous section as a *sequence* that begins with the demonstration of a single policy and continues with successive rounds of policy reuse (Fig. 3.2).

3.3.1 Policy Development and Evaluation

The sequential policy development occurs as follows. An initial policy behavior is demonstrated via teleoperation by a human teacher, and the resulting policy is refined using tactile corrections. Beginning with the demonstrated policy, successive policy behaviors then are bootstrapped from existing policies, by first employing tactile feedback for reuse in order to generate a new behavior, and following this with refinement to improve the behavior.

The demonstrated policy consists of positioning the robot end-effector to grasp the ball. A policy able to grasp the cylinder is then bootstrapped from the ball policy, which requires a new end-effector orientation. A bimanual behavior to grasp a tray is developed next, in two phases. First a policy for the right arm is bootstrapped from the cylinder policy, which requires a shift in end-effector orientation and position. The learned right-arm policy is then mirrored on the left arm. In summary, eight policies are developed for evaluation (Fig. 3.2):

π_b, π'_b : Ball grasping, derived from 4 teleoperation demonstrations (π_b) and then refined with tactile feedback (π'_b).

π_c, π'_c : Cylinder grasping, bootstrapped from the reuse (π_c) of ball policy π'_b and then refined (π'_c).

π_r, π'_r : Tray grasping for the right arm, bootstrapped from the reuse (π_r) of cylinder policy π'_c and then refined (π'_r).

π_l, π'_l : Tray grasping for the left arm, bootstrapped from mirroring (π_l) the right arm tray policy π'_r, and then refined (π'_l).

We refer to a single instance of learning this complete sequence of tasks as a *learning trial*. Three learning trials were performed for our empirical validations.

3.3.2 Successful Policy Reuse

Prior to receiving tactile feedback for the purpose of one-shot reuse, none of the original policies were able to perform the adapted tasks.

That is, the success rate of the ball policy π'_b attempting to grasp the cylinder was 0%, as was the success rate of the cylinder policy π'_c attempting to grasp the tray.

Following however tactile feedback and policy derivation according to the TPC update rule for reuse, the success rate of the adapted policies improved respectively from 0% to $85.71 \pm 10.35\%$ and from 0% to $88.89 \pm 8.05\%$ (Tbl. 3.4, $\pi_{i=c,l}$). Successful policy reuse thus was enabled through tactile feedback. Furthermore, the tactile corrections provided for refinement, following reuse, again resulted in improved policy performance. Note also that for the tray behavior, mirroring the right-tray policy on the left hand has a higher success rate than reusing the cylinder policy, which is unsurprising given similarity between the left and right tray behaviors.

Table 3.4 Performance results of sequential reuse, average of 3 learning trials.

	Mean (%)	Extremums (%)		Random (%)	Full set S (%)
		Position S_p	Orientation S_o		
π_b	100 ± 0	38.9 ± 9.6	70.8 ± 7.2	88.9 ± 19.3	68.3 ± 5.5
π'_b	100 ± 0	88.9 ± 9.6	100 ± 0	100 ± 0	96.8 ± 2.8
π_c	100 ± 0	72.2 ± 9.6	83.3 ± 14.4	100 ± 0	85.7 ± 4.8
π'_c	100 ± 0	88.9 ± 9.6	92.6 ± 7.2	100 ± 0	93.7 ± 2.8
π_r	100 ± 0	66.7 ± 0	95.8 ± 7.2	100 ± 0	88.9 ± 2.8
π'_r	100 ± 0	88.9 ± 9.6	95.8 ± 7.2	100 ± 0	95.2 ± 4.8
π_l	100 ± 0	83.3 ± 16.7	91.7 ± 14.4	100 ± 0	92.1 ± 9.9
π'_l	100 ± 0	77.8 ± 9.6	100 ± 0	100 ± 0	93.7 ± 2.8

3.3.3 Adapting the Covariance Envelope

The amount of allowable variability in a policy behavior differed between the tasks, as well as the execution dimensions. For example, compared to the ball policy from which it is bootstrapped, the cylinder policy allowed for increased variability along the principal axis of the cylinder, corresponding to the position of the hand *on* the cylinder. End-effector orientation was more constrained, however, as the palm of the hand must roughly align with the cylinder axis.

To realize the differences in acceptable variability between reused policies, during refinement tactile corrections were employed to indicate areas of desirable flexibility. Table 3.5 presents the ellipsoid level

of the covariance envelope at the final position, before and after tactile refinement. The comparatively low ellipsoid level of the ball policies reflects the absence of a flexible position dimension. That tactile refinement was able to indicate flexibility along the axis of the cylinder is shown by an increase in ellipsoid level ($1.6 \pm 0.4 \rightarrow 2.9 \pm 0.5$).

In Figure 3.3 we see that the variability learned for cylinder-grasping (a, front and side views) then was successfully preserved by one-shot reuse when adapted for tray-grasping (b, bottom and side views). In particular, the elongated envelope dimension now lies along the edge of the tray, corresponding to flexibility with respect to the position of the hand on the tray. The preservation of the covariance envelope shape, paired with the adaptation of its placement in space, is a direct result of the TPC mechanism for policy reuse. The preservation of desired variability is further confirmed by the high ellipsoid level of the cylinder being maintained in the adaptation from cylinder-grasping to tray-grasping (Tbl. 3.5, cylinder, after refinement $\rightarrow$ tray-right, before refinement).

Tactile refinement also might produce data that causes the regression envelope to narrow, in order to reflect portions of the target motion for which more precision is required. Figure 3.4 presents example trajectories for each task behavior following reuse (*Before refinement*) and then refinement (*After refinement*), where the covariance envelopes (or rather, the dimensions within Cartesian space, i.e. $\hat{\Sigma}_x$) of the final end-effector positions are shown as mesh ellipses. Images of the robot at different phases of performing each task, and from various starting positions, are also provided. For all behaviors, refinement did indeed reduce variability, with one notable exception: refinement of the cylinder-

Table 3.5 Changes in covariance envelope (within the position dimensions, $\hat{\Sigma}_x$) with refinement, average of 3 learning trials.

	Ellipsoid Level	
	Before refinement (π_i)	After refinement (π_i')
ball	2.1 ± 0.3	1.6 ± 0.2
cylinder	1.6 ± 0.4	2.9 ± 0.5
tray, right	3.4 ± 0.7	4.4 ± 1.2
tray, left	4.4 ± 1.2	4.8 ± 1.1

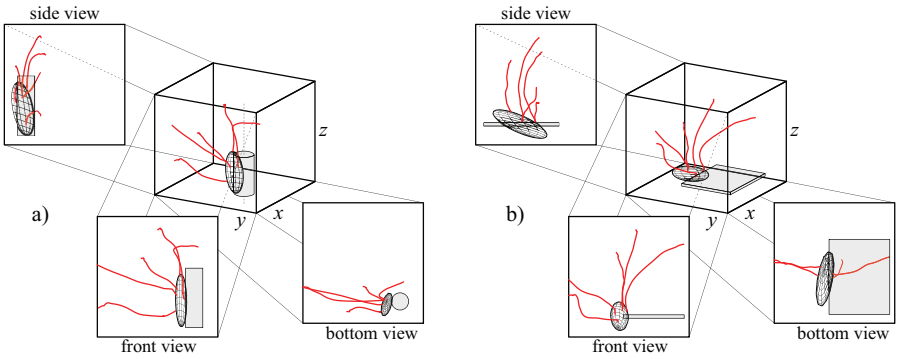

Fig. 3.3 Changes in covariance envelope (within the position dimensions, $\hat{\Sigma}_x$) with reuse. Cylinder-grasping (a) is adapted via reuse for tray-grasping (b). Callouts for each 3-D plot show a single dimension projected onto the other two dimensions. Example reproduction trajectories shown in red.

grasping policy, for which increased variance along the cylinder axis was permitted and desired.

3.3.4 Comparison to Demonstration

Polices developed under the TPC technique of reuse perform similarly to policies developed via demonstration, and so the absence of demonstration data for a specific behavior does not appear to negatively impact policy performance. The trend continues following refinement, with the TPC reuse policies producing similar or superior performance to those that received more teleoperation demonstrations.

In particular, for the cylinder policy no difference is seen between the two approaches overall (Tbl. 3.4 π_c vs. Tbl. 3.2 π_c^{4d}, Full set S). We do however note that reuse outperforms teleoperation on the position extremums (S_p), while the inverse is true for the orientation extremums (S_q); the probable explanation is that hand orientation is more constrainted for the cylinder than the ball, since the hand must align with the cylinder's principle axis while for the ball no such alignment is required. For a policy built from the reuse of the ball behavior, this constraint therefore must be indicated through refinement. For the right-hand tray-grasping policy, reuse outperforms teleoperation in all measures (Tbl. 3.4 π_r vs. Tbl. 3.2 π_r^{4d}). In this case the behaviors

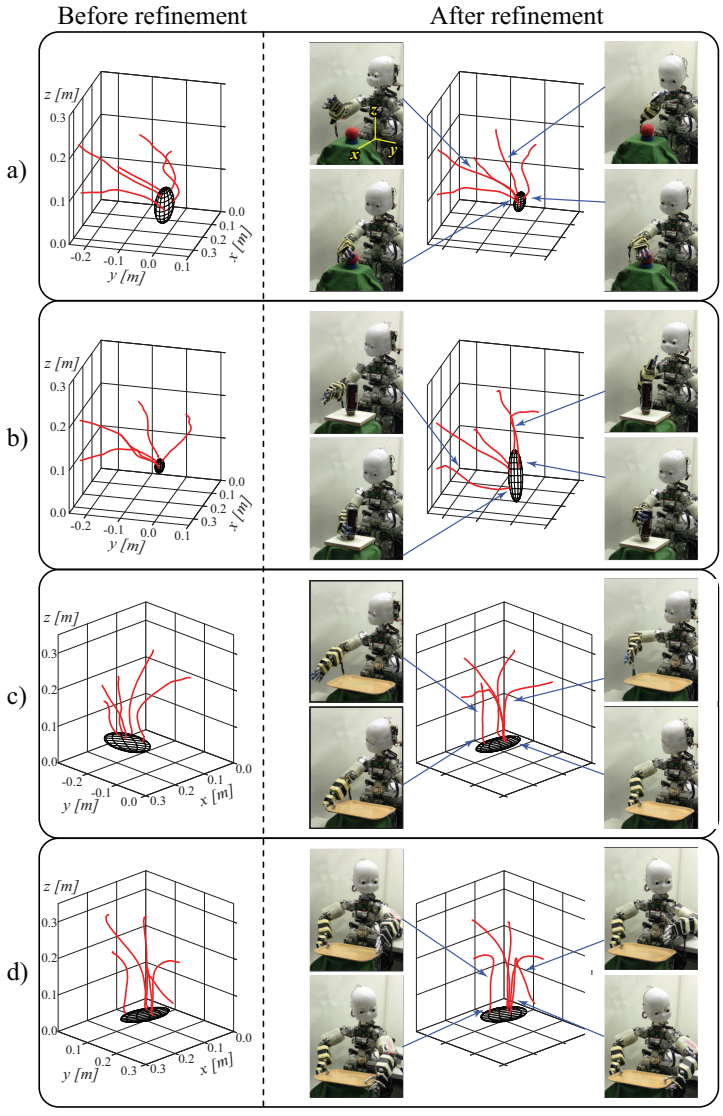

Fig. 3.4 Changes in covariance envelope (within the position dimensions, $\hat{\Sigma}_x$) with refinement, for the ball (a), cylinder (b), tray-right (c) and tray-left (d) end-effector positioning policies. Example reproduction trajectories shown in red.

were particularly well-suited for adaptation via reuse. More specifically, cylinder-grasping is flexible with respect to where the hand is placed

on the cylinder, while tray-grasping allows for variability in the position of the hand along the edge of the tray. The adapted policy in this case benefits from the preservation of variability (of covariance envelope shape), that is adapted (shifted in position and orientation) to be appropriate for tray grasping.

3.4 Reuse: Inefficient Sequence

The previous section noted that the sequence chosen for policy development was particularly well-suited for reuse. In particular, a minimal amount of covariance adaptation via refinement was required: in the sequence of ball→cylinder→tray,right→tray,left the elongated covariance envelope was learned once for cylinder-grasping, and then preserved for tray-grasping with the right and left hands. To examine the dependence of policy reuse on the selection of a suitable learning sequence, in this experiment policy development follows a sequence which we expect will be less efficient in the context of reuse: tray,right→ball→cylinder.

3.4.1 Policy Development and Evaluation

In detail, the adaptation sequence consists of demonstrated end-effector positioning to grasp a tray with the right hand, which is refined and then reused to position for ball grasping. The refined ball-grasping policy is then reused to position for cylinder-grasping, with refinement following. We expect this sequence to be inefficient with respect to covariance adaptation: in particular, that the elongated covariance envelope learned for tray-grasping will be unlearned for ball-grasping, and then relearned for cylinder-grasping. In summary, six policies are developed for evaluation:

π_r, π'_r : Tray grasping for the right arm, derived from 4 teleoperation demonstrations (π_r) and then refined with tactile feedback (π'_r).

π_b, π'_b : Ball grasping, bootstrapped from the reuse (π_b) of tray-right policy π'_r and then refined (π'_b).

π_c, π'_c : Cylinder grasping, bootstrapped from the reuse (π_c) of ball

policy π'_b and then refined (π'_c).

We again refer to a single instance of learning this complete sequence of tasks as a *learning trial*, and performed three learning trials for our empirical validations. Each of the six policies for each learning trial were evaluated on the 21 test positions in S, as defined in Section 3.1.1. A different human teacher from that of the previous sequence furthermore was employed to provide tactile corrections for learning the current sequence.

3.4.2 Policy Performance

The ability to learn successful policies for each behavior, in spite of the presumably suboptimal sequencing, was confirmed. Performance details are provided in Table 3.6.

Similar performance was seen from the tray behavior, which here was demonstrated but in the efficient sequence resulted from multiple rounds of reuse, again suggesting that policies do not suffer as a result of having no explicit demonstrations of their target behavior. The opposite is suggested by the ball behavior however, which prior to refinement did have better performance when demonstrated versus reused. We conclude therefore that sequencing order can indeed play a role in the success of reused policies. These results suggest in particular that a sequencing for which subsequent policies require broadening the covariance, rather than restricting it, is more sound. A deficit in performance however may be made up at least in part with refining

Table 3.6 Performance results of sequential reuse, inefficient sequence, average of 3 learning trials.

	Mean (%)	Extremums (%)		Random (%)	Full set S (%)
		Position S_p	Orientation S_o		
π_r	100 ± 0	61.1 ± 25.5	79.2 ± 7.2	100 ± 0	91.0 ± 8.3
π'_r	100 ± 0	77.8 ± 9.6	100 ± 0	100 ± 0	93.7 ± 3.8
π_b	66.7 ± 57.8	50.0 ± 16.7	54.2 ± 31.5	66.7 ± 33.3	57.1 ± 26.5
π'_b	100 ± 0	83.33 ± 16.7	83.3 ± 14.4	88.9 ± 19.2	85.7 ± 14.3
π_c	100 ± 0	83.3 ± 0	100 ± 0	100 ± 0	95.2 ± 0
π'_c	100 ± 0	91.7 ± 11.8	100 ± 0	100 ± 0	97.6 ± 3.4

corrections, and the ball behavior in this sequence saw a larger relative improvement in performance following refinement than the inefficient sequence (50.1% vs. 41.7% improvement).

The initial performance of the reused ball policy to accomplish the cylinder behavior was surprisingly high ($95.2 \pm 0\%$); higher than in the efficient sequence ($85.7 \pm 4.8\%$), in which the ball policy also was reused for the cylinder behavior. One possible explanation is simply that different demonstration and correction styles produce different policies, since a different human teacher was employed for the development of each sequence. A further possibility, supported by the results of the next section, is that in this sequence the covariance envelope was already appropriately constrained following reuse with respect to the location of the cylinder, and so policy performance did not suffer as much from imprecise positioning.

3.4.3 Adapting the Covariance Envelope

The evolution of ellipsoid levels (Tbl. 3.7) was less clear to interpret overall than that of the efficient sequence. The ellipsoid level increased with refinement for the tray behavior, which was expected given the results and discussion of Section 3.3.3. The absence of change in the cylinder policy similarly was not surprising given that the initial ellipsoid level is already quite high. That the ellipsoid level increased for the ball behavior however, and furthermore that this added flexibility was paired not with a decrease, but rather an increase, in performance success, was not expected.

In the previous sections we proposed that, unlike the cylinder and tray policies, the ball behavior did not have a flexible dimension along which positional variability was acceptable. In truth however there are

Table 3.7 Changes in covariance envelope (within the position dimensions, $\hat{\Sigma}_x$) with refinement, inefficient sequence, average of 3 learning trials.

	Ellipsoid Level	
	Before refinement (π_i)	After refinement (π_i')
tray, right	3.9 ± 9.0	6.5 ± 11.2
ball	6.5 ± 4.1	9.9 ± 7.4
cylinder	7.5 ± 5.2	7.5 ± 5.5

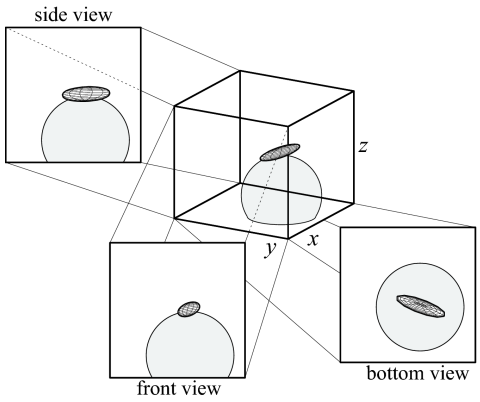

Fig. 3.5 Changes in covariance envelope (within the position dimensions, $\hat{\Sigma}_x$) with reuse, ball object within the unintuitive sequence. Callouts for the 3-D plot show a single dimension projected onto the other two dimensions. Note that the foam ball is compressed when contacted by the end-effector.

arguably two such flexible dimensions, since the hand may be positioned to have initial contact with the ball over a spectrum of positions and still successfully grasp the object, ranging from the inside to outside of the palm and the bottom of the palm to the fingertips. The teacher of the efficient sequence did not exploit either of these dimensions during demonstration or correction, preferring instead to demonstrate consistent positioning behavior. By contrast, the human teacher of the inefficient sequence exploited the palm-fingertips dimension (Fig. 3.5). These results again emphasize that differing amounts of variability can be acceptable in different dimensions, and that to increase policy performance might not in fact require an increase in precision.

3.5 Experimental Results and Setup 2: Bimanual Relative Positioning

For our second set of validation tasks, the robot learns to position both end-effectors of its 7-DoF arms for bimanual object interaction. Executions begin with the robot holding a basket in its right hand and object in its left hand. The task is then to position the basket to be in front of the robot, and position the object so that it might

be dropped into the basket. The position of the right end-effector is defined within the robot-centric coordinate frame, while the position of the left end-effector is defined within a coordinate frame centered on the right end-effector.

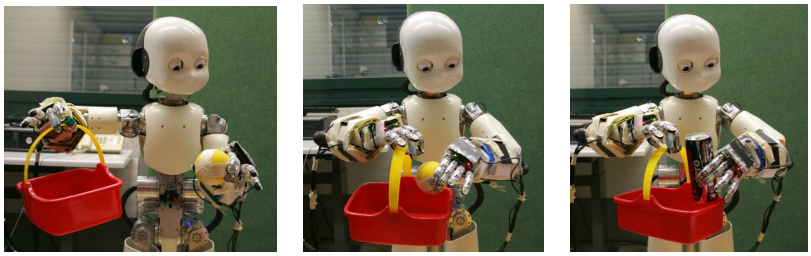

Fig. 3.6 Bimanual task of placing an object into a basket, demonstrated with a ball (left) and reused with a cylinder (right).

The robot was provided with 4 teleoperation demonstrations that placed a ball into the basket. The learned bimanual ball-basket policy then was reused to place a cylinder into the basket, whose elongated body required more clearance when being placed into the basket, as well as a change in hand orientation. Tactile corrections were provided on 2 executions with the cylinder-basket policy, constituting refinement.

Figure 3.7 plots the regression signals following both reuse (top) and refinement (bottom) for the left and right arms (average over dimensions $\hat{\Sigma}_x \in \mathbb{R}^3$ and $\hat{\Sigma}_q \in \mathbb{R}^4$). Indeed, we observe that corrections induced a large shift in orientation when the ball-basket policy is reused for the cylinder object, about midway through the task execution (red line). Corrections that then refined the cylinder-basket behavior encouraged this orientation shift to occur even earlier in the execution (yellow line) and to a more extreme degree (green line). Though no real change in position was required for the new behavior, the position of the left arm was slightly perturbed as a result of providing the tactile corrections during reuse (upper left plot). These perturbations were smoothed out following refinement however (lower left plot). Finally, note that the right arm received no corrections during reuse, since its behavior of positioning the basket to be in front of the robot is nominally the same for both objects, and so the regression signal of the right arm was unchanged by reuse.

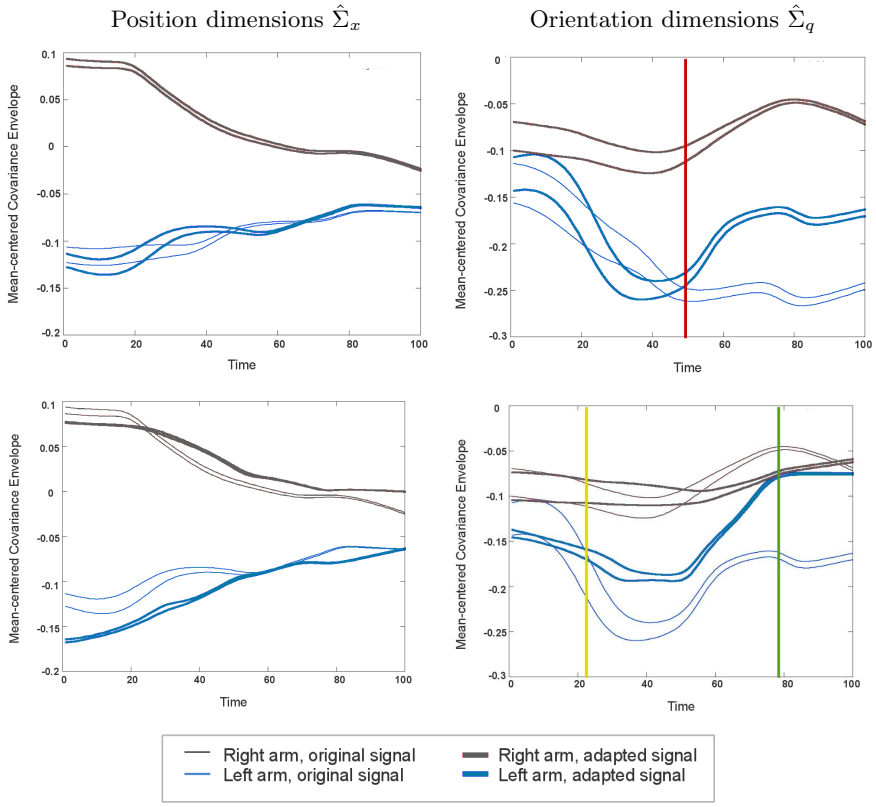

Fig. 3.7 Mean-centered covariance envelopes of the bimanual behavior modified by reuse (top) and refinement (bottom) for the left (blue) and gray (red) arms, averaged over the position dimensions $\hat{\Sigma}_x$ (left) and orientation dimensions $\hat{\Sigma}_q$ (right) of the regression prediction space. Original envelopes as thin lines, post-adaptation envelopes as thick lines.

Figure 3.8 reports on the relative change in covariance envelope with tactile corrections. The top graphs plot the (normalized) difference in covariance at each timestep before and after policy reuse, within the position (left) and orientation (right) dimensions (average over dimensions $\hat{\Sigma}_x \in \mathbb{R}^3$ and $\hat{\Sigma}_q \in \mathbb{R}^4$). Recall that the right arm received no corrections, and so there accordingly was no change in its covariance envelope (dashed line). We see however that the covariance of the left arm (solid line) holding the cylinder broadens (change in covariance > 0) within the position dimensions to facilitate a larger clearance over the side of the basket (middle peak around timestep 50, red line).

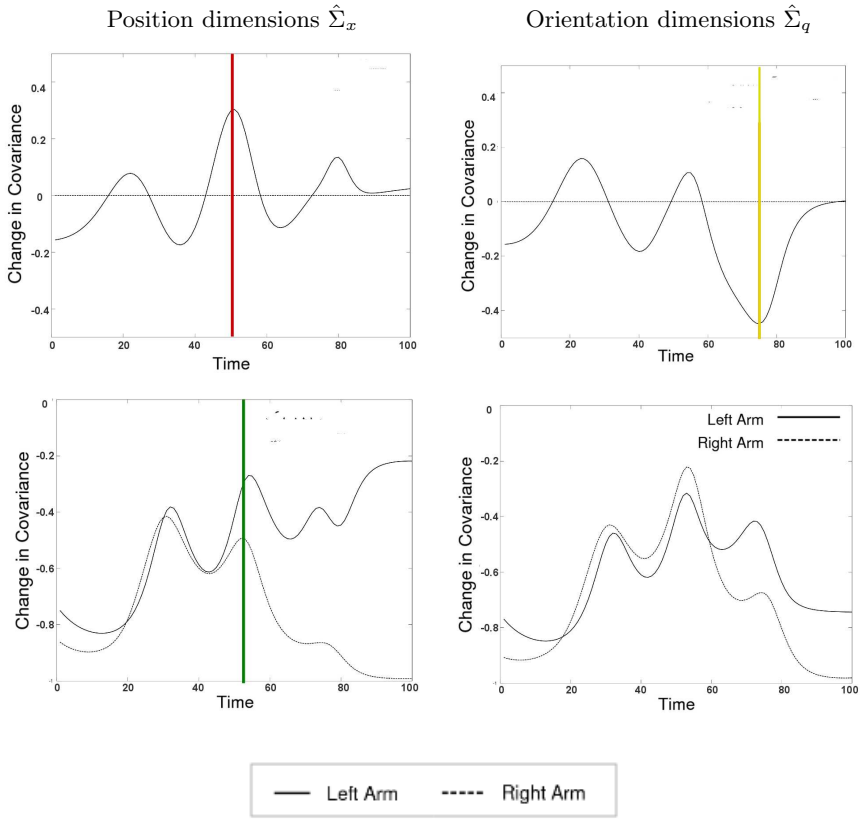

Fig. 3.8 Relative change in covariance envelope with bimanual reuse (top) and subsequent refinement (bottom), averaged over the position dimensions $\hat{\Sigma}_x$ (left) and orientation dimensions $\hat{\Sigma}_q$ (right) of the regression prediction space.

Following this, a narrowing (change in covariance < 0) of the envelope within the orientation dimensions was seen, reflecting the need for a more precise object orientation when entering the basket (large valley around timestep 75, yellow line).

The bottom graphs plot the (normalized) difference in covariance before and after refinement of the cylinder-basket policy. Within all dimensions and for both arms, the covariance envelope at each timestep was narrowed (change in covariance < 0). The positioning of the right arm (dashed line) when entering the basket was a particular target for correction, as reflected in the extreme reduction in covariance within

the position dimensions during the second half of the policy execution (after the green line).

These results confirm that the covariance of the learned policy was both narrowed and broadened at different points of the execution to facilitate adaptation to a new task. Moreover, the initial adaptation that resulted from policy reuse was further encouraged with refinement.

4

Discussion and Conclusions

The empirical results have confirmed the successful reuse and refinement of policies using tactile feedback. Here we provide discussion on key aspects of the TPC algorithm, and follow with concluding remarks.

4.1 Discussion

We begin with a discussion of tactile corrections and policy reuse as employed in this work, noting particular advantages of each. A discussion also is provided about the presence of variability within the learned policy, and the choice of weight formulation for corrected datapoints. Following this, some promising directions for future research are highlighted.

4.1.1 Tactile Corrections

There are many potential sources for suboptimal demonstrations. While the teleoperation interface employed for demonstration in this work does allow for control of a high-DoF robot arm, there are limitations. Since the robot arm is controlled by the human moving her own arm, the issue of correspondence was present, though transparent

from the perspective of the robot. Differences in correspondence instead are adjusted for online by the human while demonstrating. This limitation therefore impacts primarily the human, who furthermore must react to how another body - the robot's body, rather than her own - executes motions and interacts with the object, possibly as a mirror image if the human faces the robot. Our approach addresses suboptimal demonstration with tactile corrections. Directly touching the robot during execution has the advantage of changing the perspective of the human, who now directly interacts with the body executing the task (the robot).

Addressing the issue of *embodiment* thus is one feature of the TPC algorithm that enables the effective transfer of information from teacher to learner. Another is the *online* nature of the feedback, which allows the teacher to provide feedback in the exact areas of the state space in need of policy modification, as they are visited by the learner. The teacher therefore is not required to revisit those states, or guess as to their identity. The algorithm capitalizes on the existence of distinct instances during an execution, or equivalently along an execution trajectory, at which the policy behavior requires modification. Rather than demonstrate a trajectory in full to provide the modified behavior information, the teacher needs only to indicate a correction at these instances. The online aspect means that corrections also target exactly those areas of the state space in need of policy improvement, which can address the issue of sparsity in the demonstration set and suboptimal datapoints.

Finally, we note that in this work tactile corrections were shown to improve the behavior of policies derived from multiple, distinct, policy development techniques. In particular, the techniques of task demonstration, policy reuse and policy mirroring were all employed for policy development. While the initial performance of each technique varied, *all* were shown to benefit from tactile correction.

4.1.2 Policy Reuse

That policy reuse is *automated* is a key strength of the TPC approach: similar characteristics between the tasks are automatically extracted

for reuse, and dissimilar ones are adapted through tactile guidance.

In these experiments, reuse involved a single execution by the robot, during which the human provided corrections. By contrast, teleoperation involved 4 executions while under the control of the teacher. Not only were the number of executions greater, but the teacher was required to be actively engaged throughout the entire execution, which is not the case for reuse when the teacher needed only to be actively engaged when providing a correction. We therefore come to the qualitative conclusion that reuse required *less effort* than teleoperation, and without a sacrifice in performance.

When examining policy refinement in the first set of experiments, it was noted that the largest improvement came from refining the sole policy that derived from teleoperation demonstrations (Tbl. 3.2, ball). The cause was the demonstrated policy's relatively low initial success rate, in comparison to those policies derived from reuse. This trend also was observed for comparisons within a single task (Tbl. 3.4), where similar or superior performance was consistently achieved through reuse in comparison to teleoperation. These results suggest that reuse is more effective at transferring domain knowledge than is teleoperation.

Admittedly these results are strongly tied to our robot platform and teleoperation mechanism, as well as to the task behaviors. Though not the case for any of the tasks under consideration in this work, presumably there exists a point at which tasks are sufficiently dissimilar for reuse to be effective, and thus when teleoperation becomes the more effective tool for transferring domain knowledge. The dissimilarity between tasks may be roughly gauged by the amount of correction required for reuse to be effective. Another consideration might be whether the new task requires that the covariance envelope be broadened versus narrowed; Section 3.4.2 posited that reuse for a behavior that requires covariance narrowing might be less efficient than broadening.

4.1.3 Reflecting Demonstration Variability in the Policy

This work employed a variant on the GMM-GMR regression formulation, that allowed for deviations from the weighted mean of the demonstrations. The goal of such a formulation was to allow for flexibility in

the resulting policy execution. A noted benefit of such flexibility is the possibility of following a more direct path to the target position. As a trade-off, potential detriments included reaching the target position less reliably however.

This formulation may equivalently be seen as using differences between demonstrations as a template by which to infer those parts of the state space in which the task permits variability in the execution. Likeminded approaches have aimed to infer the crucial aspects of task execution by extracting what is similar between multiple demonstrations or demonstrators (e.g. Calinon et al. (2009); Jäkel et al. (2010); Kaiser et al. (1995); Pook and Ballard (1993)).

We highlight that, in the work of this article, acceptable variability in the task execution was effectively conveyed by the teacher through multiple modalities; namely, teleoperation and tactile corrections. Moreover, we claim that the modalities were individually better suited for different areas of the state space. In particular, to indicate generality in starting position, teleoperation was very effective. To provide generality over starting positions with tactile feedback we expect would have been quite tedious in comparison, as the tactile interface is best suited for small iterative movements. By contrast, to indicate generality at the target position was best provided through the tactile interface, which was more responsive to precise positioning.

4.1.4 Weighting New Datapoints

We also employed the idea of demonstration variability within our weight formulation for new datapoints during policy refinement. In particular, in areas that exhibited little variability during teacher demonstration, the new behavior examples produced as a result of tactile corrections were considered to be very significant. By contrast, in areas that exhibited much variability during demonstration, the presence of additional variability in the form of new corrected behavior examples was more expected, and thus considered to be less significant.

We expect the development of suitable weight functions for corrected datapoints to be an active area for future research. Many formulations are potential candidates, and their suitability depends at a

higher level on what the designer wants to see come out of the learning. For example, a separate weighting function might be employed for refinement versus reuse, instead of the one-shot formulation employed in this work. Another learning objective could be to infer the worth of particular datapoints, according to some utility function, and therefore not rely on the assumption that corrected datapoints are better examples (than the demonstrated datapoints) of the target task behavior.

4.1.5 Future Work

There are many promising extensions to this work. From an algorithmic standpoint, one might consider alternative paradigms for setting the weight on the influence of new data on a policy update, as previously discussed. Correcting within the action space is another area of interest, where for example human touch indicates changes in joint speed instead of, or in addition to, changes in pose. Such a formulation would no longer require that the policy execution be split into two parts (pose prediction and action selection), though undoubtedly would introduce nontrivial considerations with respect to implementation.

From an implementation standpoint, to validate TCP on a more sophisticated tactile sensor, that provides a richer set of feedback signals, is one direction that we are actively pursuing. Another direction is to expand the application influence of the tactile corrections, for example to correct the entire arm pose in addition to end-effector position. The formulation for policy derivation also might be improved, for example by using a dynamical systems formulation that removes time-dependence and allows for greater generalization over the state space (e.g. Khansari-Zadeh and Billard (2010)). Such a formulation furthermore would be amenable to providing corrections within the action space. The formulation for policy *re*derivation is a topic for potential future work as well. The need to keep around all of the training data is a drawback of our current system, that could be addressed by a formulation that iteratively adapts, instead of completely retrains, the learned model. Partial retraining is another option, where the model is adapted only in those areas of the state space where corrections occurred.

4.2 Conclusions

We have introduced *Tactile Policy Correction (TPC)* as an algorithm for the refinement and reuse of policies through tactile feedback from a human teacher. With tactile corrections, we aimed to improve the performance of a demonstrated behavior in response to execution experience, and to mitigate some potential limitations in demonstration-based learning. Multiple teaching modalities - namely, teleoperation and tactile corrections - were employed to provide examples of behavior execution, and we have highlighted the differing suitability of each for providing information about acceptable variability in the task behavior at different points during the task execution.

We have validated TPC on a humanoid performing end-effector positioning tasks. Tactile corrections were found to improve the performance of, and thus *refine*, a demonstrated policy. Furthermore, tactile feedback was shown to enable policy development bootstrapped from an existing behavior, and thus policy *reuse*. Comparisons to policies derived from solely teleoperation demonstration confirmed policy reuse to be an effective mechanism for transferring domain knowledge, and policy refinement to be more successful at improving performance. Future work will consider alternate algorithmic formulations for tactile refinement and reuse, and furthermore will validate TPC with a more sophisticated tactile sensor.

5

Acknowledgments

The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 - Challenge 2 - Cognitive Systems, Interaction, Robotics - under grant agreement n^o [231500]-[ROBOSKIN].

References

P. Abbeel and A. Y. Ng. Exploration and apprenticeship learning in reinforcement learning. In *Proceedings of the 22nd International Conference on Machine Learning (ICML'05)*, Bonn, Germany, 2005.

B. Argall, S. Chernova, B. Browning, and M. Veloso. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57 (5):469–483, 2009.

B. D. Argall. *Learning Mobile Robot Motion Control from Demonstration and Corrective Feedback*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, March 2009.

B. D. Argall, E. L. Sauser, and A. G. Billard. Tactile guidance for policy refinement and reuse. In *9th IEEE International Conference on Development and Learning (ICDL '10)*, Ann Arbor, Michigan, USA, 2010.

P. Baerlocher and R. Boulic. An inverse kinematics architecture enforcing an arbitrary number of strict priority levels. *International Journal of Computer Graphics*, 20, 2004.

J. A. Bagnell and J. G. Schneider. Autonomous helicopter control using reinforcement learning policy search methods. In *Proceedings of the IEEE International Conference on Robotics and Automation*

(ICRA'01), Seoul, Korea, 2001.

D. C. Bentivegna. *Learning from Observation Using Primitives*. PhD thesis, College of Computing, Georgia Institute of Technology, Atlanta, GA, July 2004.

A. Billard, S. Callinon, R. Dillmann, and S. Schaal. Robot programming by demonstration. In B. Siciliano and O. Khatib, editors, *Handbook of Robotics*, chapter 59. Springer, New York, NY, USA, 2008.

S. Calinon and A. Billard. Incremental learning of gestures by imitation in a humanoid robot. In *Proceedings of the 2nd ACM/IEEE International Conference on Human-Robot Interaction (HRI'07)*, Arlington, Virginia, USA, 2007.

S. Calinon, F. D'halluin, D. G. Caldwell, and A. Billard. Handling of multiple constraints and motion alternatives in a robot programming by demonstration framework. In *Proceedings of the IEEE-RAS International Conference on Humanoids Robots*, Paris, France, 2009.

S. Chernova and M. Veloso. Learning equivalent action choices from demonstration. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'08)*, Nice, France, 2008.

D. Cohn, Z. Ghahramani, and M. Jordan. Active learning with statistical models. *Artificial Intelligence Research*, 4:129–145, 1996.

D. H. Grollman and O. C. Jenkins. Dogged learning for robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'07)*, Rome, Italy, 2007.

G. Grunwald, G. Schreiber, A. Albu-Chaffer, and G. Hirzinger. Programming by touch: The different way of human-robot interaction. *IEEE Transactions on Industrial Electronics*, 50(4), 2003.

F. Guenter, M. Hersch, S. Calinon, and A. Billard. Reinforcement learning for imitating constrained reaching movements. *RSJ Advanced Robotics, Special Issue on Imitative Robots*, 21:1521–1544, 2007.

R. Jäkel, S. R. Schmidt-Rohr, M. Lösch, and R. Dillmann. Representation and constrained planning of manipulation strategies in the context of programming by demonstration. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '10)*, Anchorage, Alaska, USA, 2010.

M. Kaiser, H. Friedrich, and R. Dillmann. Obtaining good performance from a bad teacher. In *Programming by Demonstration vs. Learning from Examples Workshop at ML'95*, Tahoe City, California, USA, 1995.

S. M. Khansari-Zadeh and A. Billard. BM: An iterative algorithm to learn stable non-linear dynamical systems with gaussian mixture models. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '10)*, Anchorage, Alaska, USA, 2010.

J. Kober and J. Peters. Learning motor primitives for robotics. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '09)*, Kobe, Japan, 2009.

T. Minato, Y. Yoshikawa, T. Noda, S. Ikemoto, H. Ishiguro, and M. Asada. CB2: A child robot with biomimetic body for cognitive developmental robotics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '07)*, San Diego, California, USA, 2007.

C. L. Nehaniv and K. Dautenhahn. The correspondence problem. In K. Dautenhahn and C. L. Nehaniv, editors, *Imitation in Animals and Artifacts*, chapter 2. MIT Press, Cambridge, MA, USA, 2002.

M. N. Nicolescu and M. J. Mataric. Methods for robot task learning: Demonstrations, generalization and practice. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'03)*, Melbourne, Victoria, Australia, 2003.

P. K. Pook and D. H. Ballard. Recognizing teleoperated manipulations. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '93)*, Atlanta, Georgia, USA, 1993.

J. Saunders, C. L. Nehaniv, and K. Dautenhahn. Teaching robots by moulding behavior and scaffolding the environment. In *First Annual Conference on Human-Robot Interactions (HRI '06)*, Salt Lake City, Utah, USA, 2006.

M. Stolle and C. G. Atkeson. Knowledge transfer using local features. In *Proceedings of IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL'07)*, Honolulu, Hawaii, USA, 2007.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA, London, England, 1998.

J. D. Sweeney and R. A. Grupen. A model of shared grasp affordances from demonstration. In *Proceedings of the IEEE-RAS International Conference on Humanoids Robots (Humanoids'07)*, Tokyo, Japan, 2007.

N. Tsagarakis, G. Metta, G. Sandini, D. Vernon, R. Beira, F. Becchi, L. Righetti, J. Santos-Victor, A. Ijspeert, M. Carrozza, and D. Caldwell. iCub: The design and realization of an open humanoid platform for cognitive and neuroscience research. *Advanced Robotics*, 21, 2007.

K. Wada and T. Shibata. Social effects of robot therapy in a care house - change of social network of the residents for two months -. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '07)*, Rome, Italy, 2007.