# Policy Adaptation through Tactile Correction

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

**Abstract.** Behavior adaptation based on execution experience can be a practical tool to increase the robustness of a robot behavior learned from demonstration. While demonstration learning is a powerful technique for the development of robot behaviors, in general development remains a challenge. 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 adaptation of a policy learned from demonstration. We provide an initial validation of refinement under the TPC algorithm on humanoid robot performing a grasp positioning task, and policy performance is found to improve with tactile corrections. We additionally show different modalities, namely teleoperation and tactile corrections, to provide information about allowable variability in the target behavior in different areas of the state space.

## 1 Introduction

Motion control is fundamental to many robotics applications, yet the development of paradigms for control remains a challenge. Difficulties such as noisy sensors and inaccurate world models, for example, often confound the development of control *policies*, that map world state to robot actions. Moreover, many of the challenges associated with policy development only grow with domain and robot complexity, for example high degree of freedom humanoids.

Policy development therefore typically involves a significant measure of expertise and effort, prompting the advancement of techniques to reduce the requirements placed on a developer. One option is to increase policy robustness through adaptation in response to execution experience. Some development effort also might be mitigated if a new policy is built by bootstrapping from an existing policy. The approach to policy development taken in this work is founded on both ideas, of policy refinement and reuse. The policy modifications furthermore are guided by human teacher feedback, a fundamental characteristic of this work.

We present *Tactile Policy Correction (TPC)* as an algorithm that incorporates human feedback in the form of tactile corrections for the purposes of policy adaptation (Fig. 1). Policy corrections are indicated through the touch of a human teacher, and the teacher provides corrections in order to accomplish both policy refinement and reuse. More specifically, our approach initially derives a policy via *Learning from Demonstration (LfD)* techniques. Under LfD, the robot generalizes a policy from data recorded during demonstration executions by a task expert. Our generalization formulation additionally produces policies that reflect the inherent *variability* seen within the teacher demonstrations, allowing for greater flexibility in the resulting behavior during learner reproduction. Policy *corrections* from a human teacher then are provided through a tactile interface. The policy predictions provide target poses for the robot body, and when a correction is indicated, the robot immediately modifies its pose to accommodate the adjustment. The resulting execution then is treated as new demonstration data for the policy.

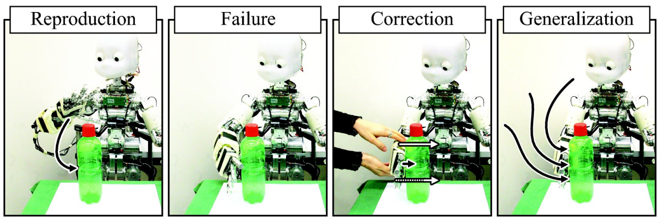

**Figure 1.** Our approach of policy adaptation via corrections provided through a tactile interface on the body of the robot. Black arrows indicate robot motion and white arrows human hand movement.

We provide results from a preliminary validation of policy refinement under TPC, with a humanoid robot performing a grasp-positioning task. We show the policies produced under our approach to both be flexible with respect to teacher demonstration variability, as well as to improve in performance when provided with tactile corrections. We additionally find that different teaching modalities (i.e. task demonstration, tactile correction) provide information about the allowable variability in task execution within different areas of the execution state space.

The following section provides related literature and motivation for our approach. The TPC algorithm is detailed in Section 3, along with our approach for preserving demonstration variability in the learned policy. Initial validation results on humanoid robot and discussion are presented in Section 4, and in Section 5 we conclude.

## 2 Background and Motivation

This section begins with a discussion of policy development and refinement under *Learning from Demonstration (LfD)*, proposes the use of tactile corrections for refinement and then delineates the motivating factors for our approach.

### 2.1 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 on a variety of robotics applications, and has the attractive characteristics of being an intuitive means for human teacher to robot

[1] Learning Algorithms and Systems Laboratory (LASA), École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, email: [brennadee.argall, eric.sauser, aude.billard]@epfl.ch

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 a 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 [2] and [5] 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 [13]. 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. [18]), while under *kinesthetic demonstration* the teacher touches the robot for control (e.g. [7]). Teleoperation requires an interface for direct robot control, and possibly also a translation by the teacher to the body of the robot; kinesthetic teaching requires a (passive or active) responsiveness to human touch. Both techniques are employed in our work.

Policy derivation then amounts to building a predictor that reproduces the demonstrated behavior. Many approaches exist within LfD to derive a policy from the demonstration data [2], 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 (RL)*. 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 teacher feedback, discussed in Section 3.1.

## 2.2 Policy Refinement and Reuse

Even with the advantages secured through demonstration, policy development typically is still a non-trivial task. Here we consider two techniques to assist the policy development process, or equivalently to reduce the strain on the policy developer: policy *refinement* and policy *reuse*. To have a robot learn from its execution performance, or *experience*, is a valuable policy improvement tool for any development technique, and execution experience may be used to refine a policy. The ability to reuse existing policies, designed to address related tasks, is also a practical feature for any policy learning system.

Within the context of LfD specifically, execution experience can be used to overcome limitations in the demonstration dataset. One typical limitation is dataset sparsity, since demonstration from every world state is infeasible in all but the simplest domains. Other limitations include the issue of correspondence between the teacher and learner or deficiencies in the performance of the teacher, who may in fact provide suboptimal or ambiguous demonstrations. Overcoming potential dataset limitations is key to good policy performance, since a LfD policy depends heavily on the quality of the demonstration data from which it is derived.

Under LfD, 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 [17] and learned state transition models [1]. Other approaches provide more demonstration data, driven by more teacher-initiated demonstrations [7] as well as by learner requests for more data [8, 9]. In this work, we also provide more data, but through a different mechanism than that which provides the initial teacher demonstrations. 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 [14], demonstrated primitives are combined into a new policy by a human [16] or automatically by the algorithm [2], and demonstrated tasks are decomposed into a library of primitives [4]. 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.[2]

## 2.3 Tactile Corrections

A fundamental consideration in our work is the mechanism used to provide feedback to the learner; for example, feedback provided verbally to indicate successful task completion [11]. The feedback form we explore is this work is that of *tactile corrections*.

To correct poor policy predictions is a particularly direct approach to addressing potential LfD limitations. In comparison to overall performance evaluations or state rewards, that can provide an indication of the quality of a policy prediction, corrections furthermore provide guidance on what might have been a more suitable alternate prediction. While more teacher demonstration can populate sparse datasets or provide an improved behavior example, it also requires visiting the state in which the policy requires improvement, which can be impractical for real world domains.

Within LfD, policy correction has seen limited attention. In most approaches a human teacher indicates the correct prediction from a discrete set of actions with significant time duration [8, 14]. Even fewer provide corrections within *continuous* state-action spaces sampled at a *rapid* rate: both characteristics of low-level motion control, which is the target application domain for our work. To accommodate these constraints, our approach translates feedback from a tactile sensor into continuous-valued modifications on the current pose as the robot executes. In contrast to other works with continuous-valued corrections [2], here corrective feedback is offered *online*, instead of post-execution, and through a *tactile interface*, instead of a high-level corrective language.

We posit that tactile feedback furthers many of the strengths of demonstration-based learning. Namely, humans use touch to instruct other humans in certain contexts, for example when demonstrating a pose or motion that requires a particular position or trajectory in 3-D space, like a ballet posture or tennis swing. To augment demonstration learning with tactile feedback therefore is one natural extension to the idea of teaching robots as humans teach other humans. Another attractive feature of demonstration-based policy development is accessibility to those who are not robotics experts. Policy development by non-experts furthermore strongly suggests robot operation around humans, in which case the detection of tactile interactions can be crucial for safe robot operation. Tactile sensing thus gains importance on a very fundamental level, and the detected tactile interactions may be exploited further for knowledge transfer from human to robot.

Within the field of *Human-Robot Interactions (HRI)*, a handful of works utilize human touch for the development of robot behaviors. For example, during humanoid behavior learning, tactile feedback is detected in order to minimize the support forces provided by a

[2] We emphasize that the initial validation of TPC provided in this paper addresses policy refinement only. When used for reuse instead of refinement, however, the algorithm remains unchanged. Rather, it is only the intent of the teacher that changes, who guides the learner motion to exhibit a different behavior from that which was demonstrated.

teacher [12], and during interactions between a robotic pet-surrogate and elderly patients, tactile reward signals are used within RL to adapt behavior selection [19].

## 2.4 Motivation for our Approach

In summary, the approach presented in this paper employs *tactile* corrections to modify a policy learned through demonstration, for the purposes of policy *refinement* and *reuse*. The factors motivating this approach include the following.

The first factor is our target application domain: low-level motion control for high DoF robots. Policy development for high DoF robots is difficult. To specify a target behavior for each joint is complicated, and systems typically are under-constrained, resulting in many joint configurations mapping to a single end-effector pose. Demonstration-based policy development thus has many advantages. Correspondence differences motivate the use of teleoperation for demonstration, though we note that from an implementation standpoint teleoperation becomes more challenging when many degrees of freedom must be controlled.

The second factor is to overcome potential limitations in the demonstration dataset. For example, suboptimal behavior examples can result from a poor interface for controlling a demonstration, and other potential drawbacks include dataset sparsity. We aim to overcome dataset limitations through policy refinement. To accomplish policy refinement, the approach presented in this paper provides new behavior examples. The source for these examples, however, is *not* more teacher demonstration. Instead, we have the student respond online to corrections indicated by a teacher, and the resultant trajectory is treated as new training data for the policy.

The final factor is how to indicate the policy corrections that result in the new behavior examples. Our approach provides corrections through a tactile interface. 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, that furthermore is relatively unaddressed within the LfD literature to date.

## 3 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.

Under TPC, a policy is initially derived from demonstrations of a task by a teacher. We formally define the world to consist of actions $A \in \mathbb{R}^{\ell}$ and observations $Z \in \mathbb{R}^{(m+n)}$ of world state. An observation $\boldsymbol{z} \in Z$ 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.[3] We define a *demonstration* to consist of a sequence of observations $\{\boldsymbol{z}^j\}_{j=1}^{\tau}$, recorded during teacher execution of the task. The collected set $D = \{\boldsymbol{z}^j\}_{j=1}^{N}$ of demonstrations is then provided to the robot learner. From this set a policy $\pi : Z \to A$ is derived, that enables the selection of an appropriate action given the observed state.

Following demonstration and policy derivation, the robot executes with its policy and receives tactile corrections from the human teacher. Tactile corrections are used within two capacities, either to refine the existing policy or to build a new policy bootstrapped on the demonstrated policy. Pseudo-code for this approach is provided in Algorithm 1.

**Algorithm 1** *Tactile Policy Correction*

```
 1: Given $D$
 2: initialize $\boldsymbol{\delta}^0 \leftarrow \mathbf{0}$
 3: derive $\pi \leftarrow$ policyDerivation($D$)
 4: while correcting do
 5:    Policy $\pi$ execution:
 6:       predict $\hat{\boldsymbol{z}}_{\varphi}^t \leftarrow$ regression$(\boldsymbol{z}^{t-1})$
 7:       execute $\boldsymbol{z}_{\varphi}^t \leftarrow$ controller$(\hat{\boldsymbol{z}}_{\varphi}^t + \boldsymbol{\delta}^t)$
 8:    if detect touch then
 9:          map $\boldsymbol{\delta}_{\epsilon}^t \leftarrow M(touch)$
10:          correct $\boldsymbol{z}_{\varphi}^t \leftarrow$ controller$(\boldsymbol{z}_{\varphi}^t + \boldsymbol{\delta}_{\epsilon}^t)$
11:          record $\boldsymbol{\delta}^{t+1} \leftarrow \boldsymbol{\delta}^t + \boldsymbol{\delta}_{\epsilon}^t$
12:    end if
13:    set $w^t$
14:    record $D \leftarrow D \cup (\boldsymbol{z}^t, w^t)$
15: end while
16: rederive $\pi \leftarrow$ policyDerivation($D$)
17: return $\pi$
```

### 3.1 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 $\pi$. The second phase of the algorithm involves learner execution with the policy $\pi$, and corrective tactile feedback which is used to update $\pi$. This *execution-correction-update* cycle continues to the satisfaction of the teacher.

A single execution-correction-update cycle is presented in lines 4-16 of Algorithm 1. Policy execution (lines 5-7) 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 6). Action selection is accomplished via a robot-specific controller, and its execution results in a new robot pose $\boldsymbol{z}_{\varphi}^t$ (line 7).

The human teacher may choose to offer a tactile correction at any timestep of an execution. If detected, the robot learner translates the tactile feedback into an incremental shift $\boldsymbol{\delta}_{\epsilon}^t \in \mathbb{R}^m$ in the robot pose, according to mapping $M$ (line 9). Note that the form taken by the tactile feedback is platform-specific, depending both on the tactile sensors employed to detect human touch 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 10). 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 11) and is added to the pose prediction at each execution timestep (line 7).

The timestep concludes with the recording of observation $\boldsymbol{z}^t$, along with a weight $w^t \in [0, 1]$ for the new datapoint (details in Sec. 3.3.2), into the set $D$ (line 14). The tactile correction thus also is recorded, since the current pose has been corrected by tactile feedback and is recorded into observation $\boldsymbol{z}^t$ through component $\boldsymbol{z}_{\varphi}^t$. Upon completion of the entire execution, policy $\pi$ is rederived from demonstration set $D$ (lines 16); the corrected execution thus is treated as new data for the policy.

Important to note is that the TPC algorithm is agnostic to the techniques used for pose prediction (`regression`) and action selection

[3] 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 $\boldsymbol{z}_{\neg\varphi} = \emptyset$.

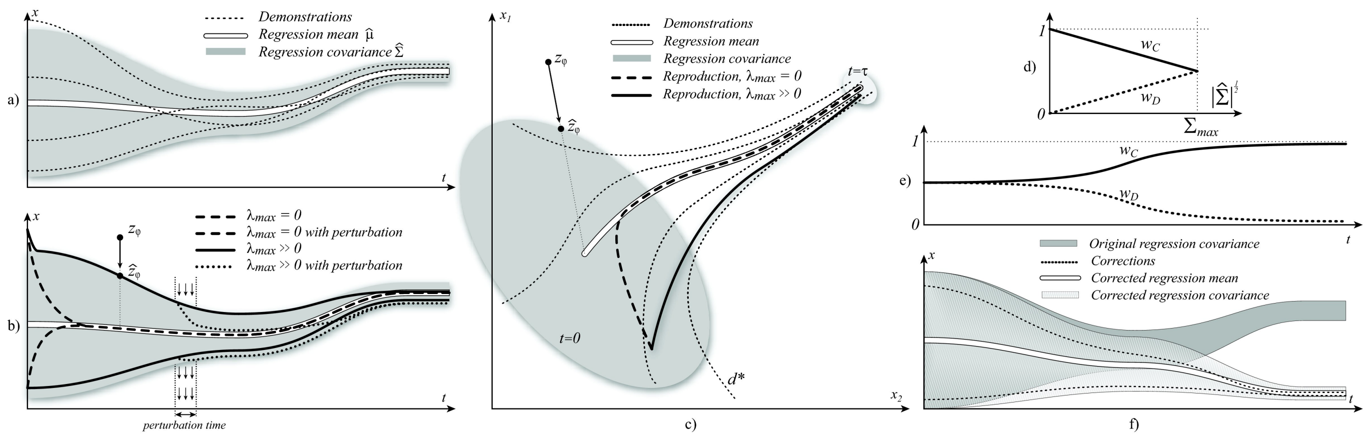

**Figure 2.** a) Demonstration data and resulting GMR regression mean and covariance envelope. b,c) Illustration of our offset formulation for GMR that allows for deviation from the regression mean, showing adaptability with respect to perturbations (b, dotted lines) and starting position (c, bold line). d) Our weight function formulation, as a function of covariance envelope size. e,f) Illustration of an example weight function and the resulting shift in regression signal.

(`controller`) during policy execution, as well as to the technique that translates tactile feedback into a pose adjustment (mapping $M$). The following sections describe the particular techniques we employ for the implementation of the TPC algorithm within this work.

## 3.2 Policy Execution

This section describes the specific techniques used for policy execution under our implementation of the TPC algorithm. For pose prediction, the *GMM-GMR* regression technique is employed (Sec. 3.2.1), with a modification to allow for variability in the resulting policy (Sec. 3.2.2). For action selection we use an inverse kinematic controller (Sec. 3.2.3).

### 3.2.1 Pose Prediction

Target poses are predicted through the *GMM-GMR* algorithm [7], which first encodes demonstrations in a *Gaussian Mixture Model (GMM)* and then predicts a target pose through *Gaussian Mixture Regression (GMR)*. The recorded demonstrations are modeled probabilistically in a GMM, whose parameters are trained under a weighted version of the *Expectation-Maximization (EM)* algorithm (details in Sec. 3.3.2). Our implementation defines observation component $\boldsymbol{z}_\varphi$ as 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 $\boldsymbol{z}_{\neg\varphi} \equiv t \in \mathbb{R}$ as the timestep of 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 $\boldsymbol{z}_{\neg\varphi}$, i.e. $p(\boldsymbol{x}, \boldsymbol{q}|t)$.

### 3.2.2 Deviating from the Mean Trajectory

We additionally take advantage of the probabilistic nature of the regression to generate variability, and thus flexibility, in the predicted trajectory. Under GMR, a target pose $\hat{\boldsymbol{z}}_\varphi^t$ is predicted with mean $\hat{\boldsymbol{\mu}}^t$ and covariance $\hat{\boldsymbol{\Sigma}}^t$ (Fig. 2a). We modify the pose prediction by

$$\hat{\boldsymbol{z}}_\varphi^t = \hat{\boldsymbol{\mu}}^t + \boldsymbol{\delta}_\lambda^t \tag{1}$$

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

$$\boldsymbol{\delta}_\lambda^t = \begin{cases} \Delta_\lambda^t & if \quad \lambda^t \le \lambda_{\max} \\ \Delta_\lambda^t \frac{\lambda_{\max}}{\lambda^t} & otherwise \end{cases} \tag{2}$$

$$\Delta_\lambda^t = \boldsymbol{z}_\varphi^t - \hat{\boldsymbol{\mu}}^t \quad , \quad \lambda^t = \|(\hat{\boldsymbol{\Sigma}}^t)^{-\frac{1}{2}} \Delta_\lambda^t\| \tag{3}$$

where $\boldsymbol{\delta}_\lambda^t$ is defined by the difference between the current robot pose and regression mean ($\Delta_\lambda^t$), and whether the magnitude ($\lambda^t$) of this difference (inversely scaled by standard deviation $(\hat{\boldsymbol{\Sigma}}^t)^{\frac{1}{2}}$) exceeds a threshold ($\lambda_{\max}$). Without this offset, the pose predictions follow exactly the mean trajectory. An example where this would not be desired is if the execution starting position is actually closer to the target position than is the start of the mean trajectory. Predicting the mean trajectory in this case will cause the learner execution to backtrack unnecessarily. With our offset, the pose predictions are no longer constrained to only follow the mean trajectory (Fig. 2b,c, $\lambda_{\max} \gg 0$). Instead, variations are allowed within the constraints of the regression covariance.

The amount of allowable deviation is dictated in terms of an acceptable number ($\lambda_{\max}$) of standard deviations from the regression mean, where $\lambda_{\max} \ge 0$ is a constant parameter set by hand (in our empirical validation, $\lambda_{\max} = 2$). For execution points (including starting positions) within this threshold (i.e. within $\lambda_{\max}$ standard deviations of the regression mean $\hat{\boldsymbol{\mu}}^t$), the execution proceeds with its current pose (i.e. $\hat{\boldsymbol{z}}_\varphi^t = \boldsymbol{\mu}^t + \Delta_\lambda^t = \boldsymbol{z}_\varphi^t$). Execution points outside of this threshold are first projected (e.g. Fig. 2c, $\boldsymbol{z}_\varphi$ 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 in the teacher demonstrations.

One gain of this regression formulation, that will be confirmed in Section 4.2, is allowing the learner execution to choose a more direct path to the goal, that perhaps deviates from the mean trajectory but is still within the bounds of what was demonstrated. Another potential gain in using offset $\boldsymbol{\delta}_\lambda^t$ is online adaptation to external perturbations, for example a compliant response to a perturbation caused by unexpected contact with a human. Consider an external force applied to the robot arm during execution, (i) to which we would like the robot to respond (ii) while still attaining the goal (e.g. Fig. 2b). Without our offset, the learner execution will attain the second objective, but not the first; that is, the learner will continue to follow the mean trajectory, and thus attain the target position, but will not be responsive to the force ($\lambda_{\max} = 0$, dashed lines). With our offset, however, both objectives are attained ($\lambda_{\max} \gg 0$, dotted lines). The offset value is the execution's response to the perturbation, and since the offset is constrained by the covariance envelope (more specifically, to keep the execution within $\lambda_{\max}$ standard deviations of the regression mean), the target position also will be attained.[4]

### *3.2.3 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 velocities $\dot{\boldsymbol{\theta}} \in \mathbb{R}^7$ for the joint angles of a robot arm. The manipulator of our implementation (Sec. 4.1) 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 $\hat{\boldsymbol{z}}_{\varphi}$ and the current robot pose $\boldsymbol{z}_{\varphi}$ by using a pseudo-inverse method that both avoids joint limits and is robust to singularities [3].

## 3.3 Tactile Corrections

The tactile interface consists of five *Ergonomic Touchpads* located on the manipulator arm. The pads detect contact presence and relative motion, which we map to a change in end-effector position and orientation (Sec. 3.3.1). The movement that results is recorded into the dataset and incorporated into a policy update (Sec. 3.3.2).

### *3.3.1 Online Modification of Policy Execution*

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. 3a,b). In practice, we decompose the mapping $M$ into two distinct parts, that operate separately on the wrist (end-effector) and on the hand of the robot arm, which seemed a more intuitive mapping for the experimenters providing corrections.[5]

The first part of the mapping operates on the first 5-DoF leading to the wrist of our 7-DoF manipulator. The pads $T_{\{0..3\}}$ can be seen as an interface for controlling the manipulator along the 6-DoF Cartesian space (position and orientation). Sliding the fingers along two opposite touchpads can either lead to a translational or rotational motion command, depending on whether the sliding directions agree or not (Figure 3c,d). When motion is detected on the touchpads, the commands are mapped to a target velocity for the end-effector in Cartesian space, and then resolved to target joint velocities by an inverse kinematic controller [3]. The second part of the mapping relates to the final 2-DoF controlling the robot hand. Since the last pad has 2-DoF as well, here the mapping to motion commands is one to one.

[4] With a reasonably-sized factor $\lambda_{\max}$ (i.e. reasonably small, such as $\lambda_{\max} = 2$), attaining the target position with our formulation depends primarily on the covariance envelope being small at the target position, which the unmodified formulation of GMR relies on as well.

[5] Touchpad feedback is somewhat limited in comparison to more sophisticated tactile sensors, for example that provide force information or a finer spatial resolution. In practice corrective repositioning is not always as responsive as the teacher requires, and so we pause policy execution such that psuedo-code lines 9-10 loop until repositioning is complete. Note that this limitation results from a deficiency in *hardware*, not the algorithm, and validation with a more sophisticated tactile sensor is a target for future work.

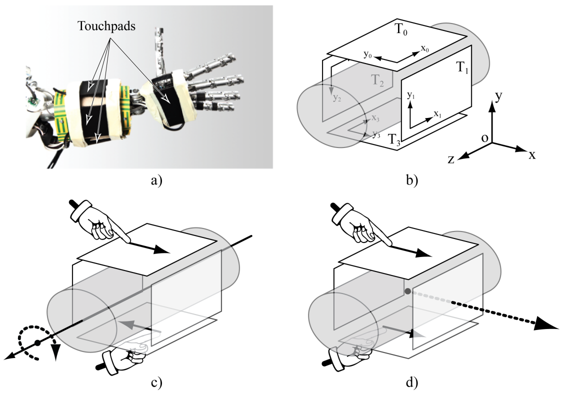

**Figure 3.** a,b) Schematic of the touch pads controlling the robot wrist and hand. c,d) Fingers sliding on opposite pads produces rotational (c) or translational (d) motions.

### *3.3.2 Incorporation into a Policy Update*

Upon completion of an execution corrected with tactile feedback, new data is incorporated into the policy. To incorporate feedback, the policy is rederived taking into account the new datapoints. Policy execution within the TPC algorithm consists of a pose prediction via regression techniques, followed by action selection by a controller. Under our implementation the controller is statically defined, and so policy derivation consists of regression parameter estimation only. Policy rederivation thus consists of re-estimating the regression parameters, which is accomplished through a weighted version of the EM algorithm (details in Tbl. 1).

**Table 1.** Weighted Expectation-Maximization (EM)

Our weighted version of the EM algorithm modifies the EM implementation of [7] to include weight $w^j$. The algorithm loops between the *E-step* and the *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*:

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

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

$$w_C(t) = 1 - \frac{|\hat{\Sigma}^t|^{\frac{1}{2}}}{2 \cdot \Sigma_{\max}} \ , \quad \Sigma_{\max} = \max_t |\hat{\Sigma}^t|^{\frac{1}{2}} \qquad (4)$$

$$w_D(t) = 1 - w_C(t) \qquad (5)$$

where $|\hat{\Sigma}^t|$ 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 ($t \equiv \boldsymbol{z}^j_{\neg\varphi}$) of the observation recording.

With this weight formulation, we assume that the 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 ($|\hat{\Sigma}^t|^{\frac{1}{2}} \ll \Sigma_{\max}$), corrected datapoints are given a high weight, and the regression signal accordingly shifts strongly. By contrast, in areas of high covariance ($|\hat{\Sigma}^t|^{\frac{1}{2}} \rightarrow \Sigma_{\max}$), it is not unexpected that executions might differ from the demonstrated behavior, and so demonstrated and corrected execution points are given approximately equal weight. This weight formulation is shown in Figure 2d, followed by an example weight function (e) and resulting shifted regression signal (f).

## 4 Empirical Validation

Here we provide initial validation results of TCP on a high DoF humanoid (Sec. 4.1), highlighting in particular the flexibility of our regression formulation (Sec. 4.2) and refinement of the demonstrated behavior (Sec. 4.3). A discussion of these results, and future research directions, are also provided (Sec. 4.4).

### 4.1 Robot Task and Domain

Initial validation of the TPC algorithm is performed with a 57-DoF humanoid robot, the *iCub* robot, performing a grasp positioning task. The task consists of positioning the robot end-effector (7-DoF) to grasp a cylindrical object. Demonstration is performed via teleoperation, accomplished with a *3D mouse* able to control all 6-DoF of the end-effector position and orientation. Closing the hand for grasping is handled by a static controller.[6] Demonstrations are provided from multiple starting end-effector positions with respect to the object. Tactile corrections are then performed using the touchpad setup described in Section 3.3.1.

The focus of this validation is two-fold: (i) to explore policy *flexibility* with respect to the variability present within the teacher demonstrations and (ii) to confirm the ability of TPC to *refine* a demonstrated behavior. The validation of policy *reuse* is left for future work. Three policies therefore are developed for comparison:

$\pi$ : Derived from the demonstration set $D$ using GMR as in [7].

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

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

[6] 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. Note also that if controlling the hand is also a part of the demonstrations, then the operational space is 15-DoF and a more complex teleoperation system is required.

### 4.2 Flexible and More Efficient Policies

Flexibility in the policy execution is enabled through offset factor $\boldsymbol{\delta}_\lambda$. As illustrated in Figure 2c, without this offset the execution trajectory (dashed line) will follow 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 demonstration set executions. 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 4 confirm this behavior with real robot data.

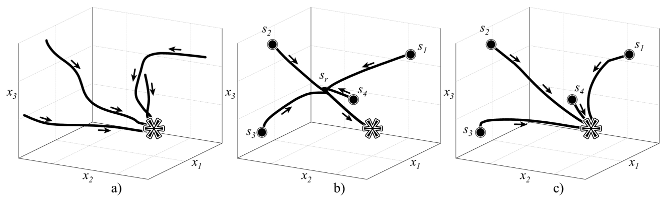

**Figure 4.** a) Demonstration executions to target position $*$. b,c) Executions from starting positions $s_1..s_4$, performed by policy $\pi$ (b) and policy $\pi_\lambda$ (c). Note that executions by policy $\pi_\lambda$ proceed directly to the target position, while those of $\pi$ first visit the start of the mean trajectory ($s_r$).

Table 2 provides the lengths of the execution trajectories (as fractions of the distance traveled by policy $\pi$) 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 through shorter trajectory lengths ($\pi_\lambda$ vs. $\pi$). The most dramatic improvement is seen with starting position $s_4$, whose position is such that the execution must travel explicitly away from the target position ($*$) to reach the start of the mean trajectory ($s_r$). In this case overt backtracking is the result if offset $\boldsymbol{\delta}_\lambda$ is not employed (Fig. 4).

**Table 2.** Execution Length (from multiple starting positions, as a fraction of the execution length of policy $\pi$)

| Starting Position | $\pi$ | $\pi_\lambda$ | $\pi_{\lambda,c}$ |
|---|---|---|---|
| $s_1$ | 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** |

In the original demonstration set $D$, variability in the starting position was present. The task also allows for some variability in the *target* position, as the hand may be positioned at various locations on the cylinder and still successfully grasp the object. Variability in target position was minimally present in the demonstration set however, as 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 (the *3D mouse*). Through tactile corrections, however, the teacher *was* able to convey variability with respect to target position.

As shown in Figure 5, following correction, the covariance envelope at the target position broadened with respect to location *on* the cylinder; that is, along the principle axis of the cylinder (dimension $x_3$). Note further that the envelope by contrast narrowed within the plane supporting the cylinder (dimensions $x_1, x_2$), appropriately indicating less flexibility with respect to the location *of* the cylinder

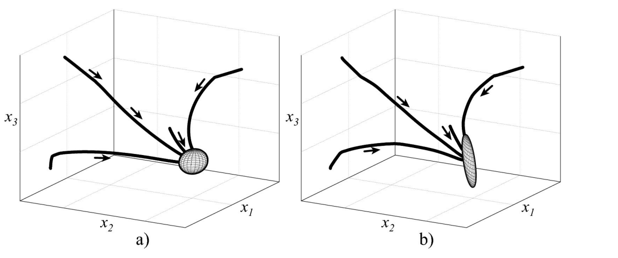

**Figure 5.** Covariance envelope (shaded region) at the target position a) before tactile corrections (i.e. with policy $\pi_\lambda$) and b) after tactile corrections (i.e. with policy $\pi_{\lambda,c}$).

(i.e. refinement). In general, allowing for additional variability at the target position resulted in similar execution lengths (Tbl. 2, $\pi_{\lambda,c}$ vs. $\pi_\lambda$), which is not unexpected given that the length of the cylinder is quite small compared to the distances between the starting and target positions. Rather, the benefit of allowing for variation at the target position is more flexible placement of the hand on the object. Flexible hand placement has many potential advantages, for example if part of the object is unreachable due to obstruction by an obstacle.

## 4.3 Refined and More Successful Behavior

Tactile corrections were able to refine the policy of the demonstrated behavior, as shown through the adaptation of the regression signal in response to the tactile feedback. For this validation task, corrections were provided near the end of the task execution, to adjust the end-effector position once it was close to the cylinder (though in theory corrections may be provided at any point during the execution). Figure 6 highlights the shift in regression envelope at the target position (discussed in the previous section), with the callouts showing the envelope to narrow in dimension $x_1$, to slightly shift in $x_2$ and to broaden in $x_3$.

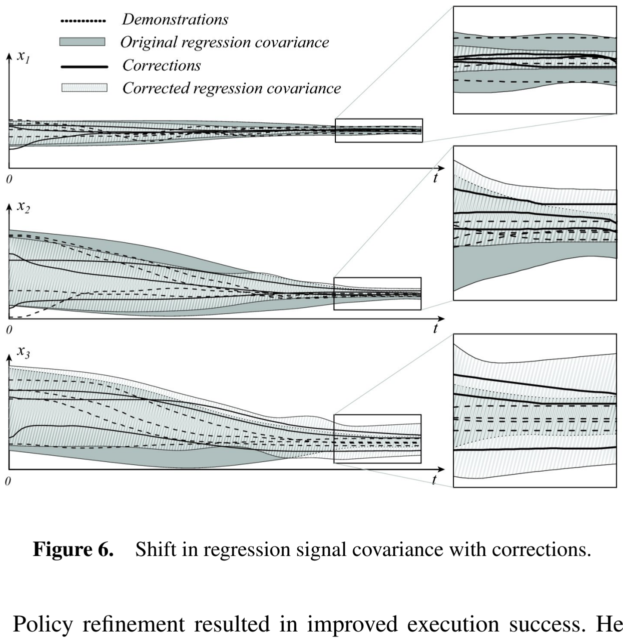

**Figure 6.** Shift in regression signal covariance with corrections.

Policy refinement resulted in improved execution success. Here success is measured by a score of 1 given to executions that successfully pick up the object, $\frac{1}{2}$ to those that grasp the object but not stably enough to prevent slipping as it is picked up, and 0 to those that are unable to grasp the object (sum of 10 executions from random starting positions). One trade-off to not following the regression mean exactly, and thus to using our offset $\boldsymbol{\delta}_\lambda$, is that executions reach the target position less reliably (Tbl. 3, $\pi_\lambda$ vs. $\pi$). Tactile corrections are able to buy back some of this lost reliability however, and thus improve policy performance. The corrected policy achieves the goal more reliably (Tbl. 3, $\pi_{\lambda,c}$ vs. $\pi_\lambda$) but with execution trajectories of similar length (Tbl. 2, $\pi_{\lambda,c}$ vs. $\pi_\lambda$), and thus without sacrificing the execution flexibility gained through offset $\boldsymbol{\delta}_\lambda$.

**Table 3.** Execution Success

| | $\pi$ | $\pi_\lambda$ | $\pi_{\lambda,c}$ |
|---|---|---|---|
| Score | 10 | 6.5 | 8.5 |

## 4.4 Discussion

Here we provide discussion on two key aspects of the approach presented in this work: the preservation of demonstration variability within learned policy, and the choice of weight formulation for corrected datapoints. We follow by highlighting some promising directions for future research.

### *4.4.1 Reflecting Demonstration Variability in the Policy*

This work employed a variant on the GMM-GMR regression formulation, that allowed for deviation from the mean trajectory of the demonstrations. The goal of such a formulation was to allow for flexibility in the resulting policy execution. Noted benefits of such flexibility included the possibility of following more direct trajectories to the target position, and online adaptability to perturbations. As a trade-off, potential detriments included reaching the target position less reliably however.

This formulation may equivalently be seen as using the differences seen 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. [6, 10, 15]).

We highlight that acceptable variability in the task execution was effectively conveyed by the teacher through multiple modalities; namely, teleoperation and tactile corrections. Moreover, 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 along a single axis at the target position was best provided through the tactile interface, which was more responsive to precise positioning.

### *4.4.2 Weighting New Datapoints*

We also employed this idea of demonstration variability within our weight formulation for new datapoints. 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.

Our weight formulation relied on the assumption that the variance present within the demonstrations matched the target variance of the adapted task. Such an assumption is reasonable for policy *refinement*, as the target behavior of the adapted policy is the same as that of the demonstrated policy. Such an assumption is less appropriate for policy *reuse*, however, when the target behavior of the adapted policy *differs* form that of the demonstrated policy, and therefore so also might the variance profile. Consider for example the demonstration of a reaching task that passes through a large hole in order to grasp an object. Now consider adapting the resultant policy to achieve the same grasping objective, but by reaching through a much smaller hole. The demonstrated task will be constrained only at the target position, where it must grasp the object. By contrast, the adapted task will be constrained additionally in the middle of the execution trajectory, when it must pass through the small hole. The comparatively large covariance in the demonstration data at this phase of the execution trajectory therefore will *not* reflect the allowed variance in the adapted task.

We therefore 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, if a separate weighting function is employed for refinement versus reuse, should the robot is to be notified when reuse is taking place, or statistically infer as much from differences between the distributions of demonstrated versus corrected datapoints? 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.

#### *4.4.3 Future Directions*

There are many promising extensions to this work. From an algorithmic standpoint, to consider alternative paradigms for setting the weight on the influence of new data in a policy update will be at the forefront of our future investigations. Already discussed is how an alternate formulation would be more appropriate for policy reuse; another possibility could be to have a distinct weighting function for each teacher, for example, as demonstrators might differ in their ability to perform the task. 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.

From an implementation standpoint, we have found tactile corrections to be effective for the purpose of refining a demonstrated behavior, and to show their effectiveness for the purposes of policy reuse is currently under development. To validate TCP on a more sophisticated tactile sensor, that provides a richer set of feedback signals, is another direction that we are actively pursuing. An additional direction for future research is to expand the application influence of the tactile corrections, for example to correct the entire arm pose instead of just end-effector position.

## 5 Conclusion

We have presented *Tactile Policy Correction (TPC)* as an algorithm for the adaptation 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. 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 provided an initial validation of policy refinement with TPC on a humanoid performing a grasp positioning task. Future work will validate TPC for policy reuse and with a more sophisticated tactile sensor.

## ACKNOWLEDGEMENTS

The research leading to these results has received funding from EC Projects IST-2004-004370 (RobotCub) and IST-04169 (feelix-growing), and from the European Community's Seventh Framework Programme FP7/2007-2013 - Challenge 2 - Cognitive Systems, Interaction, Robotics - under grant agreement n [231500]-[ROBOSKIN].

## REFERENCES

[1] Pieter Abbeel and Andrew Y. Ng, 'Exploration and apprenticeship learning in reinforcement learning', in *Proceedings of ICML*, (2005).

[2] Brenna D. Argall, *Learning Mobile Robot Motion Control from Demonstration and Corrective Feedback*, Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, March 2009.

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

[4] Darrin C. Bentivegna, *Learning from Observation Using Primitives*, Ph.D. dissertation, College of Computing, Georgia Institute of Technology, Atlanta, GA, July 2004.

[5] Aude Billard, Sylvain Callinon, Rudiger Dillmann, and Stefan Schaal, 'Robot programming by demonstration', in *Handbook of Robotics*, eds., B. Siciliano and O. Khatib, Springer, New York, NY, USA, (2008).

[6] 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 Humanoids*, (2009).

[7] Sylvain Calinon and Aude Billard, 'Incremental learning of gestures by imitation in a humanoid robot', in *Proceedings of HRI*, (2007).

[8] Sonia Chernova and Manuela Veloso, 'Learning equivalent action choices from demonstration', in *Proceedings of IROS*, (2008).

[9] Daniel H. Grollman and Odest Chadwicke Jenkins, 'Dogged learning for robots', in *Proceedings of ICRA*, (2007).

[10] Michael Kaiser, Holger Friedrich, and Rudiger Dillmann, 'Obtaining good performance from a bad teacher', in *Programming by Demonstration vs. Learning from Examples Workshop at ML'95*, (1995).

[11] Andrea Lockerd and Cynthia Breazeal, 'Tutelage and sociallyl guided robot learning', in *Proceedings of IROS*, (2004).

[12] Takashi Minato, Yuichiro Yoshikawa, Tomoyuki Noda, Shuhei Ikemoto, Hiroshi Ishiguro, and Minoru Asada, 'CB2: A child robot with biomimetic body for cognitive developmental robotics', in *Proceedings of IROS*, (2007).

[13] Chrystopher L. Nehaniv and Kerstin Dautenhahn, 'The correspondence problem', in *Imitation in Animals and Artifacts*, eds., Kerstin Dautenhahn and Chrystopher L. Nehaniv, chapter 2, MIT Press, Cambridge, MA, USA, (2002).

[14] Monica N. Nicolescu and Maja J. Mataric, 'Methods for robot task learning: Demonstrations, generalization and practice', in *Proceedings of AAMAS*, (2003).

[15] P. K. Pook and D. H. Ballard, 'Recognizing teleoperated manipulations', in *Proceedings of ICRA '93*, (1993).

[16] Joe Saunders, Chrystopher L. Nehaniv, and Kerstin Dautenhahn, 'Teaching robots by moulding behavior and scaffolding the environment', in *Proceedings of HRI*, (2006).

[17] Martin Stolle and Christopher G. Atkeson, 'Knowledge transfer using local features', in *Proceedings of ADPRL*, (2007).

[18] John D. Sweeney and Roderic A. Grupen, 'A model of shared grasp affordances from demonstration', in *Proceedings of Humanoids*, (2007).

[19] Kazuyoshi Wada and Takanori Shibata, 'Social effects of robot therapy in a care house - change of social network of the residents for two months -', in *Proceedings of ICRA*, (2007).