Tactile Guidance for Policy Refinement and Reuse

Brenna D. Argall, Eric L. Sauser and Aude G. Billard
Learning Algorithms and Systems Laboratory (LASA)
École Polytechnique Fédérale de Lausanne (EPFL), 1015 Lausanne, Switzerland
Email: [brennadee.argall,eric.sauser,aude.billard]@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 can degrade policy performance. This work presents an approach for policy improvement and adaptation through a *tactile* interface located on the body of a 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. We validate TPC on a 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.**

***Index Terms*—Imitation and Social Learning, Skill Acquisition, Human-Robot Interaction, Humanoid Robots**

I. Introduction

The challenge of motion control is fundamental to many robotics applications. One representation for control is a *policy*, that maps sensor observations of the world to robot actions. Policy execution in real environments is confounded by noisy sensors, approximate mappings and execution uncertainty. Such challenges 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. To adapt a policy based on execution experience, however, can reduce the requirements placed on a developer. With this work, we introduce an algorithm that incorporates human feedback - specifically, *tactile corrections* - for the purpose of policy adaptation. Policy corrections are indicated through the touch of a human teacher, and the teacher provides corrections in order to accomplish two goals. The first is to *refine* a policy during execution, and thus improve performance by increasing policy robustness. The second is to assist in policy *reuse*, by guiding an existing policy towards accomplishing a different task.

Our approach initially derives a policy via *Learning from Demonstration (LfD)* techniques, and then provides corrections through a tactile interface (Fig. 1). Under LfD, the teacher demonstrates a target behavior, and the robot generalizes a policy from data recorded during the teacher executions. We posit that tactile feedback furthers the idea of teaching robots as humans teach other humans, which is one strength of demonstration-based learning. Moreover, tactile detection can be crucial for safe robot operation around humans, and further exploited for knowledge transfer from human to robot. In our work the policy predictions provide target poses for the robot body. Through the tactile interface, the human teacher indicates relative adjustments to the robot pose (policy predictions) online, as the robot executes. The robot immediately modifies its pose to accommodate the adjustment, and the resulting, adjusted, pose is treated as new training data for the policy.

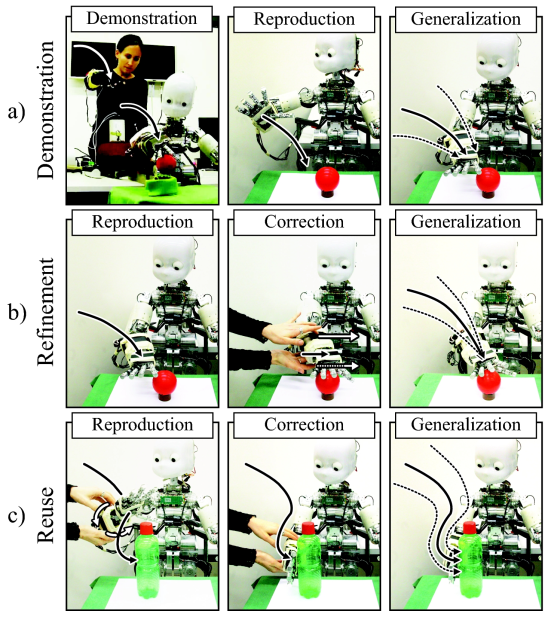


Fig. 1. Our approach of a) task demonstration, followed by tactile correction of the learned policy for b) refinement of the demonstrated behavior and c) its reuse in the development of other policies. Black solid arrows indicate demonstrated or corrected executions, black dashed arrows generalization executions and white arrows human hand movement.

We validate our approach on a humanoid robot performing grasp-positioning tasks. The performance of a policy learned from demonstration is found to improve after *refinement* from tactile corrections. Successful policy *reuse* also is shown. Through tactile guidance, executions with the demonstration policy are iteratively adjusted towards targeting a new object-location combination, resulting in a policy able to execute the alternate, undemonstrated, combination. Tactile corrections thus *enable* the development of a new policy, bootstrapped on the reuse of a policy learned from demonstration.

The following section presents related literature that supports our work. Section III details our approach, including the algorithm and tactile correction technique. Empirical validation is described in Section IV, with results provided in Section V. In Section VI, we conclude.

II. Background and Motivation

This section discusses policy derivation and adaptation within *Learning from Demonstration (LfD)*, and the indication of policy corrections via tactile feedback.

Under LfD, teacher executions of a desired behavior are recorded and a policy is derived from the resultant dataset; we refer the reader to [1] and [2] for a full review. Though demonstration has many attractive characteristics for both teacher and learner, policy development still typically is non-trivial. Potential limitations include dataset sparsity, since demonstration from every world state is infeasible in all but the simplest domains, and *correspondence issues* [3], where physical differences between the teacher and learner bodies complicate the use of the demonstration data. To minimize correspondence issues, our work accomplishes demonstration by teleoperating the robot learner as it records from its own sensors. We then derive a policy via function approximation of the mapping from world observations to robot actions.

To assist the policy development process, our work employs two policy adaptation techniques: refinement and reuse. Policy *refinement* within LfD uses execution experience to update state transition models [4] and reward-determined state values [5]; other approaches provide more demonstration data, driven by learner requests [6], [7] or more teacher-initiated demonstrations [8]. To correct poor policy predictions is a particularly direct, though relatively unexplored, approach to refinement, and in most cases a human teacher indicates the correct prediction from a discrete set of actions [6], [9]. By contrast, our approach provides continuous-valued corrections, and differs from related work [10] by offering corrective feedback *online*, instead of post-execution, and through a *tactile interface*, instead of a high-level corrective language.

Given the challenge of developing robust control policies, the ability to reuse existing policies, designed to address related tasks, is a practical feature for any policy learning system. Policy *reuse* under LfD occurs most frequently in the form of behavior primitives, or simpler policies that contribute to the execution of a more complex policy. Examples include hand-coded primitives used within [9] or automatically extracted from [11] demonstrated tasks, and primitives learned from demonstration [10]. In our work, policy reuse takes a bootstrapping, rather than behavior-primitives, form: the adapted policy performs a *different* task than the demonstrated policy, with the algorithm automatically extracting similar characteristics between the tasks.

Our approach uses *tactile corrections* from a human teacher for both policy refinement and reuse. We propose that human touch, like demonstration, is an intuitive and effective mechanism for the transfer of knowledge from human to robot. A handful of robot learning works (including but not limited to LfD) use human touch for robot behavior development. For example, tactile feedback is detected to minimize the support forces provided by a teacher during humanoid behavior learning [12], and the selection of hand-developed behaviors is adapted using tactile reward signals [13].

III. The Tactile Policy Correction Algorithm

We introduce *Tactile Policy Correction (TPC)* as an algorithm for the refinement and reuse of motion control policies through tactile feedback from a human teacher. Pseudo-code for this approach is provided in Algorithm 1.

Algorithm 1 *Tactile Policy Correction*

```
1:  Given $D$
2:  initialize $D_c \leftarrow \{\}$, $\boldsymbol{\delta}^{t=0} \leftarrow \mathbf{0}$
3:  derive $\pi \leftarrow$ policyDerivation$(D, D_c)$
4:  while correcting do
5:      predict $\hat{\boldsymbol{z}}^t_\varphi \leftarrow$ regression$(\boldsymbol{z}^{t-1})$
6:      execute $\boldsymbol{z}^t_\varphi \leftarrow$ controller$(\hat{\boldsymbol{z}}^t_\varphi + \boldsymbol{\delta}^{t-1})$
7:      if detect touch then
8:          map $\boldsymbol{\delta}^t_\epsilon \leftarrow M(touch)$
9:          correct $\boldsymbol{z}^t_\varphi \leftarrow$ controller$(\boldsymbol{z}^t_\varphi + \boldsymbol{\delta}^t_\epsilon)$
10:         record $\boldsymbol{\delta}^t \leftarrow \boldsymbol{\delta}^{t-1} + \boldsymbol{\delta}^t_\epsilon$
11:     end if
12:     set $w^t$
13:     record $D_c \leftarrow D_c \cup (\boldsymbol{z}^t, w^t)$
14: end while
15: if refine then
16:     rederive $\pi \leftarrow$ policyDerivation$(D, D_c)$
17:     return $\pi$
18: else if reuse then
19:     derive $\pi' \leftarrow$ policyDerivation$(D, D_c)$
20:     return $\pi'$
21: end if
```

A. Algorithm Execution

The first phase of the TPC algorithm consists of task demonstration by the teacher (Fig. 1a). 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. The collected set $D = \{\boldsymbol{z}^j\}_{j=1}^{N}$ of N datapoints from multiple 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.

The second phase of the algorithm involves learner execution with the policy π, and corrective tactile feedback which is used to adapt π (Fig. 1b,c). This cycle of *execution-correction-adaptation* continues to the satisfaction of the teacher, and proceeds as follows. Policy execution (lines 5-6) at timestep t consists of two phases: prediction of a target pose $\hat{\boldsymbol{z}}^t_\varphi$, 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 5). Action selection is accomplished via a robot-specific controller, and its execution results in a new robot pose $\boldsymbol{z}^t_\varphi$ (line 6). The human teacher may choose

[1]Pose information is necessary when providing corrections under TPC, and so $\boldsymbol{z}_\varphi \neq \emptyset$; for this same reason, policy execution is split into two steps - prediction of a target pose and action selection by an external controller to accomplish that pose. The presence of additional observation information is application-dependent, and possibly absent such that $\boldsymbol{z}_{\neg\varphi} = \emptyset$.

to offer a tactile correction at any timestep of an execution. If detected, the robot learner translates the tactile feedback[2] into an incremental shift $\boldsymbol{\delta}_\epsilon^t \in \mathbb{R}^m$ in the robot pose, according to mapping M (line 8). The robot controller is then passed the new *corrected* pose, modified by the incremental shift $\boldsymbol{\delta}_\epsilon^t$ (line 9). 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 10) and is added to the pose prediction at each timestep (line 6).

The timestep concludes with the recording of observation $\boldsymbol{z}^t$ into the set of corrected execution points D_c (line 13). If a tactile correction was provided, it too is recorded within observation $\boldsymbol{z}^t$, through component $\boldsymbol{z}_\varphi^t$ that encodes the current (corrected) pose. A weight w^t is associated with the recorded point, such that the policy derivation will give greater importance to the corrected data (details in Sec. III-C2). Upon completion of the entire execution, the policy is adapted from demonstration set D *and* the set of corrected executions D_c; the corrected execution thus is treated as a new demonstration for the policy. In the case of refinement, the existing policy π is replaced with an updated version via rederivation (line 16). In the case of reuse, a new policy π' is derived, leaving the original policy π unchanged (line 19).

Important to note is that the TPC algorithm is agnostic 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 describe the particular techniques we employ for the implementation of TPC presented within this work.

B. Policy Execution

This section describes the techniques used for policy execution under our specific implementation of the TPC algorithm.

1) Pose Prediction: Target poses are predicted through the *GMM-GMR* algorithm [8], 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 as a mixture of K Gaussian components with means μ_k and covariance Σ_k, $k \in 1..K$, whose parameters are trained under the *Expectation-Maximization (EM)* algorithm. 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 $\boldsymbol{z}_{\neg\varphi} \equiv \tau \in \mathbb{R}^1$ 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. $p(\boldsymbol{x}, \boldsymbol{q}|\tau)$. We additionally take advantage of the probabilistic nature of the regression to generate variability in the predicted trajectory, by applying to the regression mean a constant offset that considers both covariance and initial pose (full details to be published in future work).

2) Action Selection: Given a target pose $\hat{\boldsymbol{z}}_\varphi$, action selection is accomplished via an inverse kinematic controller. Action space A consists of joint angle velocities $\dot{\boldsymbol{\theta}} \in \mathbb{R}^7$ for a robot arm. The manipulator of our implementation (Sec. IV) is redundant, as the number of degrees of freedom (7) exceeds the number of constraints (6, end-effector 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 [14].

C. 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 changes in end-effector position and orientation (Sec. III-C1). The pose that results is recorded into the demonstration set and incorporated into a policy update (Sec. III-C2).

1) Online Modification of the 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. 2a,b). Touch data from pad $T_k, k = 0..4$, consists of a 2-D relative change in pixels $(\Delta u_k^t, \Delta v_k^t)$. 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 velocities 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.

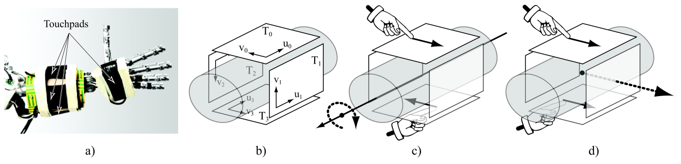


Fig. 2. Schematic of the touch pads controlling the robot wrist and hand. Fingers sliding on opposite pads results in rotational or translational motions.

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 opposite touchpads leads either to a translational or rotational motion command, depending on whether the sliding directions agree or not (Fig. 2c,d). The velocity $\dot{\boldsymbol{z}}_\varphi^t$ 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

[2]Note that the form taken by the feedback is platform-specific, depending on both the employed tactile sensors and how the sensor data is processed.

through rotation matrix R:

$$\dot{\boldsymbol{z}}_{\varphi}^{t} = \begin{bmatrix} R & \\ & R \end{bmatrix} \begin{bmatrix} \kappa_\nu \left(-\Delta v_0^t + \Delta v_2^t\right) \\ \kappa_\nu \left(\Delta v_1^t - \Delta v_3^t\right) \\ \kappa_\nu \left(-\Delta u_0^t - \Delta u_1^t - \Delta u_2^t - \Delta u_3^t\right) \\ \kappa_\omega \left(-\Delta u_0^t + \Delta u_2^t\right) \\ \kappa_\omega \left(\Delta u_1^t - \Delta u_3^t\right) \\ \kappa_\omega \left(\Delta v_0^t + \Delta v_1^t + \Delta v_2^t + \Delta v_3^t\right) \end{bmatrix}$$

Constant parameters κ_ν and κ_ω scale respectively the translational and rotational components of the touch data, to account for units differences (pixels for tactile feedback, $\frac{m}{s}$ and $\frac{rad}{s}$ for the velocity components). The mapping from Cartesian-space velocity $\dot{\boldsymbol{z}}_{\varphi}^{t}$ to joint velocity $\dot{\boldsymbol{\theta}}_{\{0..5\}}^{t}$ for the first 5-DoFs in the arm then is computed using inverse kinematics [14].

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

2) Incorporation into a Policy Update: Upon the completion of an execution that has been corrected with tactile feedback, the new data is incorporated into the policy. A weight (Alg. 1, line 12) is associated with each datapoint, depending on whether it belongs to the original (D) or corrected (D_c) dataset, such that a higher weight is assigned to *correction* points. The re-estimation of the regression parameters is accomplished through a weighted version of the EM algorithm (details in Fig. 3). Note that if the behavior induced by corrections is sufficiently different from that of the original policy, to preferentially weight the new data effectively amounts to forgetting the original behavior. A new policy therefore is instantiated for the case of reuse, when policies' behaviors differ, but not for refinement, when the target behaviors of the original and adapted policies match.

Datapoint weights are assigned based on membership in the sets of corrected points D_c or demonstrated points D. Weight w^j for point $\boldsymbol{z}^j$ is computed as

$$w^j = \begin{cases} \left(1 - \frac{N}{N+N_c}\right)(1 - \bar{w}(\tau)) & \boldsymbol{z}^j \in D \\ \left(1 - \frac{N_c}{N+N_c}\right)\bar{w}(\tau) & \boldsymbol{z}^j \in D_c \end{cases}$$

where N is the number of datapoints in D, N_c is the number in D_c and $\bar{w}(\tau)$ is a global weight function (initialized to zero). Following a corrective execution, $\bar{w}(\tau)$ updates according to

$$\bar{w}(\tau) \leftarrow \frac{1}{N_s + 1}\left(H(\tau - \tau_c) + N_s\ \bar{w}(\tau)\right)$$

where $H(\tau)$ is the Heaviside unit step function, τ_c is the time of the first correction and N_s is the number of full execution sequences previously corrected.

Our **weighted EM algorithm** modifies standard EM 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:

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

$$E_k^{(i+1)} = \Sigma_{j=1}^{N} w^j p_{k,j}^{(i+1)}$$

M-step:

$$\gamma_k^{(i+1)} = \frac{E_k^{(i+1)}}{\Sigma_{j=1}^{N} w^j}$$

$$\mu_k^{(i+1)} = \frac{\Sigma_{j=1}^{N} w^j p_{k,j}^{(i+1)} \boldsymbol{z}^j}{E_k^{(i+1)}}$$

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

Fig. 3. Data weighting and weighted Expectation-Maximization (EM).

IV. Experimental Setup

We have implemented the TPC algorithm on a small 53-DoF humanoid, the iCub robot. For our validation task, the robot learns to position[3] the end-effector of its 7-DoF arm for the grasping of different objects at various locations. Policy development consists initially of task demonstration, followed by tactile corrections to refine that policy and build other policies able to accomplish alternate tasks. In particular, policies able to arrive at a different location, or with a different orientation, are bootstrapped from the demonstrated policy.

A. Task Demonstration and Tactile Corrections

Demonstration is performed via teleoperation by a human teacher, which is non-trivial as it requires simultaneous control of all 7-DoFs of the arm. 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. 1a). Sensing units from the commercial $XSens$ joint recording 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, and then map to the joint angles of the robot arm.

This demonstration technique does allow for teleoperation of a high-DoF robot arm, but there are limitations. Since the robot arm is controlled by the human moving her arm, the issue of correspondence is 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 his own - executes motions and interacts with the object, possibly as a mirror image if the human faces the robot.

Each of these factors can result in suboptimal demonstrations, which our approach addresses with tactile corrections provided through the touch pad interface described in Section III-C1. Online corrections target exactly those areas of the state space in need of policy improvement, and directly touching the robot during execution has the advantage of

[3]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. Closing the hand for grasping thus is handled by a static controller.

changing the perspective of the human, who now directly interacts with the body executing the task. Tactile feedback furthermore may be used to bootstrap a new policy from an existing policy, thus avoiding redundant demonstration.

B. Policies and Evaluation Metrics

The demonstrations position the learner end-effector to grasp an object located at a particular position[4] within the robot-centric coordinate frame. The demonstrated policy is refined, and furthermore is reused within the development of two additional policies: positioning to grasp the object located at *different position*, which requires guidance to a new end-effector position, and positioning to grasp a *different object*, which requires guidance for a new end-effector orientation. Three distinct position-object combinations (Fig. 4) are learned, and four policies developed for evaluation:

- **Policy** $Demo$: Derived from teacher demonstration of grasping the ball object at position $P0$.
- **Policy** $Ball_{P0}$: Refinement of policy $Demo$.
- **Policy** $Ball_{P1}$: Reuse of policy $Demo$ to grasp the ball object at position $P1$.
- **Policy** $Bottle$: Reuse of policy $Demo$ to grasp the bottle object at position $P0$.

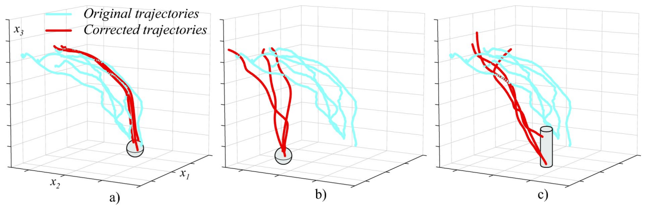
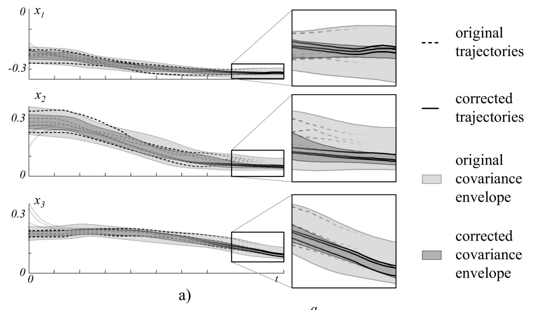


Fig. 4. Original and corrected trajectories in 3-D Cartesian space, for policies a) $Ball_{P0}$, b) $Ball_{P1}$ and c) $Bottle_{P0}$, overlaid with object illustrations.

Policies are evaluated for: *success* as the percentage of attempts that lift the object from the table, *contact number* as the number of fingers[5] in contact with the object, and *precision* improvement as the ratio of the covariance envelopes for the demonstrated vs. adapted policies. The covariance envelope provides an indication of precision since a policy's pose predictions are constrained by the boundaries of this envelope. Contact number was visually observed by the experimenters, since the iCub fingertips do not have pressure sensors.

V. Empirical Results

Both policy refinement and reuse were accomplished via tactile correction under the TPC algorithm. Policy refinement resulted in improved grasp success and contact number, while reuse enabled the development of policies able to execute undemonstrated position-object combinations. Five demonstrations were provided to policy $Demo$, and three execution-correction-adaptation cycles occurred for each policy developed through corrective techniques (policies $Ball_{P0}$, $Ball_{P1}$, $Bottle$). Each policy was evaluated by executing from 15 random starting positions (*reproduction executions*).



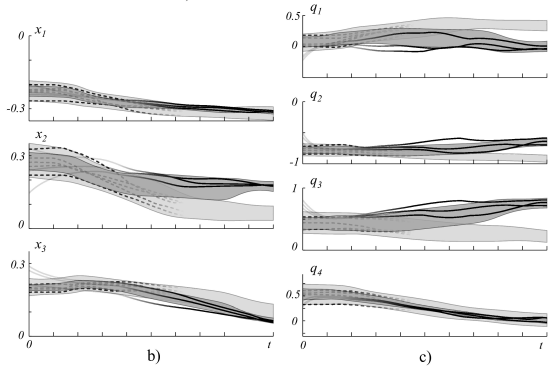


Fig. 5. Demonstrated and corrected execution trajectories for a) policy $Ball_{P0}$, b) policy $Ball_{P1}$ and c) policy $Bottle_{P0}$, with covariance envelopes. The lines boldness of the trajectoriess varies with the global weight function $\bar{w}(\tau)$ (s.t. black:$\bar{w}(\tau) = 1$, white:$\bar{w}(\tau) = 0$).

A. Performance Improvement with Tactile Corrections

Policy refinement was seen through an improvement in execution success and contact number (Tbl. I, policy $Demo$ vs. $Ball_{P0,r0}..Ball_{P0,r2}$, i.e. the policies resulting after each execution-correction-adaptation cycle). Tactile refinement adjusted the final end-effector position, and accordingly the weight (Fig. 5, line boldness) on corrected points was heavier at the end of the execution, following the correction, while the weight on demonstrated points was heavier at the start. Refinement also improved execution precision, seen through a decrease in regression envelope size within the Cartesian components of the robot pose (Fig. 5a). Note however that the precision improvement declined as a result of the variability introduced with further correction executions (Tbl. I, $Ball_{P0,r1}$ vs. $Ball_{P0,r2}, Ball_{P0,r3}$).

TABLE I
Policy Refinement

Policy	$Demo$	$Ball_{P0,r0}$	$Ball_{P0,r1}$	$Ball_{P0,r2}$
Success	86.7	100	100	93.3
Contact	1.7 ± 1.0	2.9 ± 0.4	2.7 ± 0.5	2.7 ± 0.8
Precision	1	1.6	1.4	1.4

[4]The validation task was intentionally formulated to be inflexible with respect to object position, in order to clearly illustrate, and thus provide proof-of-concept validation of, reuse and refinement under TPC. Simply shifting the coordinate frame to be object-centric, instead of robot-centric, removes this restriction and thus enables generalization with respect to object position.

[5]It is possible, but less stable, to pick up an object with fewer than 3 fingers in contact (we consider only 3 of the 4 fingers, as 2 fingers of the hand are coupled). Grasps that fail to pick up the object have a contact number of 0.

B. Tactile Feedback Enables Successful Policy Reuse

Policy reuse was seen with the successful execution of undemonstrated behaviors $Ball_{P1}$ and $Bottle$ (Tbl. II). Note that policy $Demo$ failed for both of these novel position-object combinations. The development of policy $Ball_{P1}$ required a shift in end-effector *position*, seen with a change in the regression envelope mean within the Cartesian components of the pose (Fig. 5b). By contrast, the development of policy $Bottle$ required a change in end-effector *orientation*, seen through a shift within the quaternion pose components (Fig. 5c).

TABLE II
POLICY REUSE

	grasping **Ball** at **P1**		grasping **Bottle** at **P0**	
Policy	$Demo$	$Ball_{P1}$	$Demo$	$Bottle$
Success	0	93.3	0	100
Contact	0	2.3 ± 0.9	0	2.7 ± 0.5

Regarding the number of *correction key points*, or distinct instances of tactile repositioning, two were required for the development of policy $Ball_{P1}$ and three for policy $Bottle$. The difference in key point number was reflected in the weights of the corrected datapoints, which for policy $Bottle$ increased in strength more rapidly (Fig. 5b vs 5c, line boldness). This trend extends to policy $Ball_{P0}$ (Fig. 5a), whose single correction key point induced the slowest weight change.

C. Discussion

The empirical results confirm the successful reuse and refinement of policies learned from demonstration. 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. Two features of the TPC algorithm particularly enable the effective transfer of information from teacher to learner. The first 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 attention, as they are visited by the learner. The teacher therefore is not required to revisit those states, or guess as to their identity. The second relates to the issue of *embodiment*, which is particularly relevant for high-DoF robots. Since the teacher indicates physical adjustments directly on the body executing the policy, no compensation for any differences in correspondence is required.

Many of the algorithm's advantages relate directly to the idea of *correction key points*: 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 the key point. Note that the number of correction key points increases with the dissimilarity between the behaviors of the new and original policies. For example, to refine the demonstrated policy required only one key point, while to reuse the ball policy to grasp a bottle required three.

There are many promising extensions to this work. Regarding the algorithm, one might consider alternative paradigms for setting the datapoint weights, for example that weight new data differently for refinement versus reuse. A mechanism to select between multiple policies, developed as a result of reuse, could extend behavior development to more complex domains. Regarding implementation, to validate TPC on a more sophisticated tactile sensor, that provides a richer set of feedback signals, is one direction being actively pursued.

VI. 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 mitigate some potential limitations in demonstration-based learning. We have validated TPC on a humanoid performing grasp-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 the demonstrated behavior, and thus policy *reuse*.

ACKNOWLEDGMENT

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^o [231500]-[ROBOSKIN].

REFERENCES

[1] B. Argall, S. Chernova, B. Browning, and M. Veloso, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.
[2] A. Billard, S. Callinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. New York, NY, USA: Springer, 2008, ch. 59.
[3] C. L. Nehaniv and K. Dautenhahn, "The correspondence problem," in *Imitation in Animals and Artifacts*, K. Dautenhahn and C. L. Nehaniv, Eds. Cambridge, MA, USA: MIT Press, 2002, ch. 2.
[4] P. Abbeel and A. Y. Ng, "Exploration and apprenticeship learning in reinforcement learning," in *Proceedings of ICML*, 2005.
[5] M. Stolle and C. G. Atkeson, "Knowledge transfer using local features," in *Proceedings of ADPRL*, 2007.
[6] S. Chernova and M. Veloso, "Learning equivalent action choices from demonstration," in *Proceedings of IROS*, 2008.
[7] D. H. Grollman and O. C. Jenkins, "Dogged learning for robots," in *Proceedings of ICRA*, 2007.
[8] S. Calinon and A. Billard, "Incremental learning of gestures by imitation in a humanoid robot," in *Proceedings of HRI*, 2007.
[9] M. Nicolescu and M. Matarić, "Methods for robot task learning: Demonstrations, generalization and practice," in *Proceedings of AAMAS*, 2003.
[10] B. D. Argall, "Learning mobile robot motion control from demonstration and corrective feedback," Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, March 2009.
[11] D. C. Bentivegna, "Learning from observation using primitives," Ph.D. dissertation, College of Computing, Georgia Institute of Technology, Atlanta, GA, July 2004.
[12] 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 IROS*, 2007.
[13] 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 ICRA*, 2007.
[14] P. Baerlocher and R. Boulic, "An inverse kinematics architecture enforcing an arbitrary number of strict priority levels," *International Journal of Computer Graphics*, vol. 20, 2004.