Learning Robot Tactile Sensing for Object Manipulation

Yevgen Chebotar[1], Oliver Kroemer[1] and Jan Peters[1]

***Abstract*— Tactile sensing is a fundamental component of object manipulation and tool handling skills. With robots entering unstructured environments, tactile feedback also becomes an important ability for robot manipulation.**

In this work, we explore how a robot can learn to use tactile sensing in object manipulation tasks. We first address the problem of in-hand object localization and adapt three pose estimation algorithms from computer vision. Second, we employ dynamic motor primitives to learn robot movements from human demonstrations and record desired tactile signal trajectories. Then, we add tactile feedback to the control loop and apply relative entropy policy search to learn the parameters of the tactile coupling. Additionally, we show how the learning of tactile feedback can be performed more efficiently by reducing the dimensionality of the tactile information through spectral clustering and principal component analysis. Our approach is implemented on a real robot, which learns to perform a scraping task with a spatula in an altered environment.

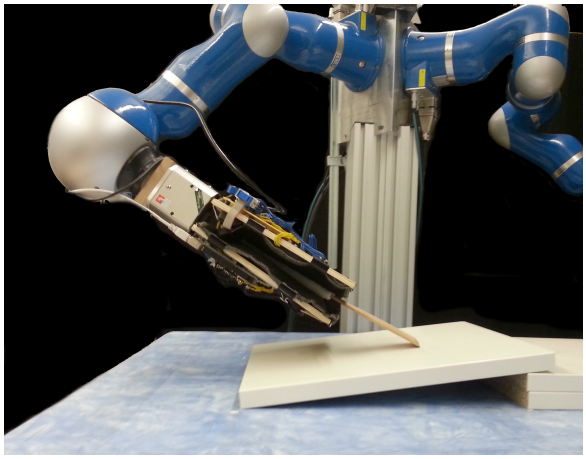

Fig. 1. Robot performing a gentle scraping task using tactile feedback.

I. Introduction

Manipulation of objects and use of tools are among the most impressive abilities of humans. The sense of touch plays a crucial role in these tasks. Moreover, impairment of the tactile sensibility leads to a significant loss of manipulation skills as tactile information is needed for performing the sensorimotor control [1]. Thus, in order to be able to use tools and assist humans in unstructured environments, it is important for robots to make use of tactile feedback. When visual information is unavailable, especially due to the in-hand occlusion, tactile information also provides additional cues about the object state, such as its position and orientation. Some of the object properties, e.g. its material or internal physical state, are often only accessible through the use of tactile perception [2].

Human manipulation control is based on the prediction of sensory information and reactions to deviations from these predictions [1]. Human hands contain four types of tactile afferent neurons: fast-adapting (FA-I, FA-II) and slow-adapting (SA-I, SA-II). The fast-adapting afferents react to the high-frequency changes of skin deformation, e.g. during an initial contact of an object with the hand or contact of an object inside the hand with another object. During a manipulation task, the fast-adapting afferents convey changes of the task phase. The slow-adapting afferents are sensitive to low-frequency stimuli and provide cues about the object position and static forces acting on it.

In this work, we use two tactile matrix arrays attached to a parallel gripper in order to acquire tactile information of grasped objects. We divide the process of using this information for manipulation into several steps. First, we localize the object inside the hand, which is important for a correct interpretation of the tactile signal during a manipulation task. Second, we learn a robot movement and collect tactile information associated with this movement. In particular, we employ dynamic motor primitives (DMPs) [3] for learning a movement from human demonstration by kinesthetic teach-in. The tactile information is encoded as a desired tactile trajectory and tactile feedback is added to the system through perceptual coupling [4]. Third, the parameters of tactile feedback are learned with the relative entropy policy search (REPS) reinforcement learning algorithm [5]. We face the problem of high dimensionality of the tactile data and therefore perform dimensionality reduction with spectral clustering [6] and principal component analysis [7]. We evaluate our method by performing a gentle scraping task using a spatula as shown in Fig. 1. Tactile feedback is used to adapt to the changes of task conditions.

The paper is organized as follows. In Section II, we describe three localization algorithms for the in-hand pose estimation. In Section III, we introduce our approach for adding tactile feedback to DMPs, dimensionality reduction of tactile data and optimization of feedback parameters. In Section IV, we present results collected during robot experiments.

A. Related work

Although tactile sensing has been an object of study for a long time, there has been little work on its use for manipulation tasks. A range of works has focused on recognizing various object properties. Object materials could

[1]Yevgen Chebotar, Oliver Kroemer and Jan Peters are with the Technische Universitaet Darmstadt, Intelligent Autonomous Systems, Germany. Jan Peters is also with the Max Planck Institute for Intelligent Systems. {chebotar,kroemer,peters}@ias.tu-darmstadt.de

be classified more accurately in [8] by learning a lower-dimensional representation of the tactile data with the help of visual information. Additionally to material properties, Chitta et al. [2] used tactile matrix arrays attached to robot fingers to derive high-frequency tactile features for determining internal states of an object, such as detection of movements and presence of a liquid inside a container.

Low-resolution tactile matrix arrays have been used for object identification by Schneider et al. [9]. A histogram codebook of appearances of different object parts was created for identifying objects based on similar part appearances. In this work, we use local features of object parts, such as intensity patches of tactile images, for the in-hand localization. Touch based perception has been studied by Petrovskaya et al. [10]. The authors created 3D object models and used readings of a touch sensor for Bayesian estimation of the object pose. They used the pose information for manipulating a box and operating a door handle. In our work, besides localization we focus on the actual dynamic tactile feedback during the task execution.

Corcoran and Platt [11] use particle filtering to model objects based on contacts with a robot hand. The object state is estimated during manipulation by integrating the likelihood of contact measurements over possible contact positions. Apart from using positive contact information, the authors also employ the information about negative contacts, i.e. where the robot hand does not touch the object, to improve localization performance. Li et al. [12] use high-resolution GelSight tactile sensor to localize objects in the hand by matching key points between object height maps with RANSAC. The localization is used to perform insertion of a USB cable dependent on the object pose.

Hsiao et al. [13] employ trajectories in the task space that depend on the pose of the manipulated object. Decision theory approach is used to choose trajectories based on the current belief of the object pose and to reduce uncertainty about the success of the manipulation. Contrary to the pose estimation in the task space, in this work we perform object localization inside the robot hand.

In contrast to the works that use tactile data for controlling a robot, Bekiroglu et al. [14] use it for assessing the stability of a grasp. The authors take into account uncertainty and study learning from single measurements and time series of measurements. Dang and Allen [15] also explore the evaluation of the grasp stability connected with the pose of the object inside the hand. After creating a database of tactile experiences with corresponding grasp stability assessments, the robot adjusts the pose of the hand such that the grasp becomes stable.

Before incorporating tactile feedback, we employ imitation learning for initializing robot movements. Imitation learning and DMPs have been extensively used in the past for performing complex movement tasks such as table tennis [16], T-ball batting [17], grasping [18], etc. The introduction of perceptual feedback by coupling of external sensor variables to control has led to improved task performance in difficult setups, e.g. in a ball-in-a-cup task [4]. In contrast to defining reward as a deviation from a static goal, such as the distance between a ball and a cup, we look at the deviation from a desired sensory trajectory. Matching of demonstrated and robot movement trajectories through reinforcement learning has shown robustness to changes of task conditions [19]. In our work, we perform matching to the desired tactile trajectory and use deviations from it to correct robot movements. The idea is also reflected in associative skill memories [20] that proposed to record stereotypical sensor signals along with the movements and use them to compute deviations from the desired behavior. However, in contrast to predefined feedback parameters, we learn optimal parameters through trial and error by applying reinforcement learning.

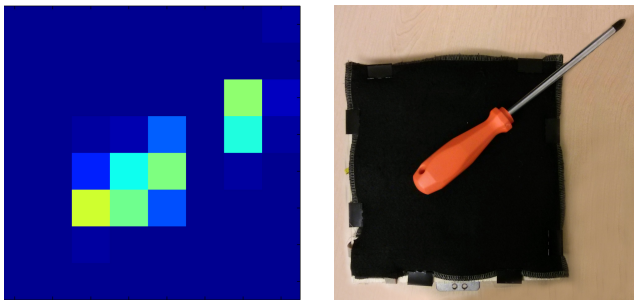

Fig. 2. Tactile image of a screwdriver handle (left) acquired from a tactile matrix array (right).

II. In-Hand Localization

In this section we describe three pose estimation algorithms for localizing objects inside the robot hand: probabilistic hierarchical object representation, iterative closest point and voting scheme. Pose estimation consists of two phases. In the first phase, a model of an object is learned from the collected tactile data. In the second phase, pose inference is performed with the help of the learned model to estimate the object pose from the new observations. Although originally developed for visual data, the presented algorithms can be used for tactile images acquired from tactile matrix arrays by performing grasps of an object. In this work, we use low-resolution tactile sensors, which imposes an additional challenge for the localization. Fig. 2 shows an example of a tactile image of a screwdriver handle (left) and a tactile matrix array (right). In all localization algorithms, we use intensity value vectors of tactile image patches as features of the object appearance.

A. Probabilistic Hierarchical Object Representation

Probabilistic hierarchical object representation (PHOR) is an approach developed by Detry et al. [21] for creating a part-based hierarchical model of an object.

An object model is a Markov tree, in which each node corresponds to a specific part of an object. The edges of the tree encode relative transformations between parent nodes and their children. They are represented as non-parametric probability distributions of possible transformation values and are denoted as *compatibility potentials*. The bottom of the tree consists of *primitive features* that correspond to

local appearances of the smallest object parts. The higher-level nodes in the hierarchy represent meta-features, i.e. object parts that are created by grouping the lower-level features. The root of the tree represents the whole object. All probability densities are encoded with kernel density estimation (KDE) [22].

The model is created iteratively in a bottom-up approach. Given a set of tactile observations, they are grouped to form bigger object parts. Spatial relationships between individual parts are preserved by encoding relative transformations in compatibility potentials on the edges of the hierarchy.

Pose inference is performed by using non-parametric belief propagation [23]. In particular, given a set of new tactile observations, the pose information is propagated from the bottom to the top of the hierarchy. Compatibility potentials are used to compute poses of the parents based on the poses of their children. Interestingly, the information can not only be propagated upwards but also downwards from parents to the children. Hence, this approach estimates poses of occluded parts of the object if we have an estimate of the pose of their parents.

B. Iterative Closest Point

Iterative closest point (ICP) is a general purpose algorithm for finding a transformation that minimizes the total distance between two point clouds [24]. Pose estimation is performed by minimizing the distance between the point cloud created from tactile observations during the model building phase to a cloud built from observations during the pose inference phase.

In the model building phase, tactile observations are transformed to a point cloud by placing a point whenever there is a tactile value greater than a specified threshold. In addition to the point coordinates, we save an appearance vector, i.e. the intensity values, of the local patch that is centered around this point. The final model consists of the points with associated appearance vectors.

Given a previously created model and a set of new tactile observations converted to a point cloud, we can run ICP for estimating the object pose. In particular, for each point in the new cloud we search for the closest point in the model cloud. Our distance measure combines the distance between point coordinates in the Cartesian space and Euclidean distance between appearance vectors associated with these points. The algorithm is repeated until the total distance between all points of both clouds falls below a specified threshold.

A disadvantage of the ICP method is that it often converges to a local optimum. In contrast, the inference in the two other algorithms is probabilistic and global convergence reduces to finding the highest mode of a distribution.

C. Voting Scheme

The approach of detecting objects and estimating their parameters in images by using voting schemes originates in the Hough transform [25]. It was first developed for identifying lines in images by transforming candidate line points in Hough space and then voting for possible slope and intercept values that could go through the single points. In this work, we take an approach similar to Glasner et al. [26]. In particular, we employ an appearance-based voting scheme where similar appearances of tactile image patches are used for collecting weighted votes for probable poses of an object.

The model of an object consists of an *appearance set* of tactile image patches and a *pose set* of relative transformations between these patches and the object frame. The relative position p_{rel} of a patch is the difference between the object position p_{obj} and the position of the patch p_{patch} in the hand plane. The relative angle θ_{rel} of a patch is the difference between the strongest gradient orientation of the patch θ_{patch} and the object angle θ_{obj} in the hand plane.

For estimating the object pose from a new set of tactile observations, we calculate appearance descriptors of each patch in the tactile image and search for its K nearest neighbors in the appearance set. Then, for each found patch, we compute a vote for the object pose based on the relative transformation p_{rel} and θ_{rel} in the pose set. Each vote is inversely weighted by the distance between appearance descriptors. Finally, we construct a non-parametric distribution of the votes with KDE. We use the mean-shift algorithm for finding all modes of the distribution [27]. The highest mode corresponds to the most probable pose of the object.

III. Tactile-Based Manipulation

After localizing an object inside the hand, we can redefine the tool center point according to the estimated object pose. In this section, we explain our method for incorporating tactile feedback into robot manipulation tasks. First, we describe dynamic motor primitives (DMPs) for encoding robot movements from human demonstration. Then, we show how tactile feedback is added to the system through perceptual coupling and how the dimensionality of the tactile information can be reduced. Finally, we explain how policy search can be used to learn parameters of the tactile coupling through trial and error.

A. Dynamic Motor Primitives

DMPs [3] are non-linear dynamical systems that consist of two components: a canonical system z and states x_1, x_2 of a spring-damper system with a forcing function f, which is driven by the canonical system. We use the following formulation of the DMPs:

$$\dot{x}_2 = \tau\alpha_x(\beta_x(g - x_1) - x_2) + \tau a f(z),$$

$$\dot{x}_1 = \tau x_2,$$

$$\dot{z} = -\tau\alpha_z z,$$

where $\alpha_z, \alpha_x, \beta_x$ are constant parameters, a is the amplitude modifier of the forcing function, τ is a time parameter for tuning speed of the movement execution and g is the goal, i.e. the final position of the movement. The value of z starts with one and approaches zero with exponential decay. The value of the forcing function also approaches zero with decreasing

z and therefore, for $z \to 0$ the spring-damper system drives the position towards the goal g.

By introducing the function $f(z)$, we add a force to the spring-damper system. Hence, by choosing a correct $f(z)$ we can encode arbitrary movements. The forcing function depends on the canonical system and is formulated as follows:

$$f(z) = \frac{\sum_{i=1}^{m} \psi_i(z) w_i z}{\sum_{i=1}^{m} \psi_i(z)},$$

$$\psi_i(z) = \exp\left(-h_i(z - c_i)^2\right),$$

where m is the number of normalized weighted Gaussian kernels with their centers distributed along the state axis z of the canonical system. At various state values z different Gaussians $\psi_i(z)$ become active and their values are multiplied by the corresponding weights w_i. The goal of the imitation learning process is to find weights w_i such that the resulting motion closely resembles the demonstration, which can be achieved with linear regression.

B. Perceptual coupling and tactile feedback

In order to react to different tactile stimuli, we have to modify the robot movement. For this task, we adapt DMPs with perceptual coupling as described by Kober et al. [4]. In particular, we define the desired tactile trajectory $\bar{\mathbf{y}}(z)$ and the current tactile signal $\mathbf{y}$. The desired tactile trajectory is recorded during the demonstration of the movement, where we record the current tactile image at each time point. Tactile feedback is based on the difference between $\bar{\mathbf{y}}(z)$ and $\mathbf{y}$. Prior to computing the feedback, tactile images have to be aligned with the images from the demonstration using a pose estimation method as described in Section II.

For encoding the desired tactile trajectory $\bar{\mathbf{y}}(z)$ and reproducing it during the task execution, we take an approach similar to Pastor et al. [20]. We model the sensory trajectory as a non-linear dynamical system in the form of a DMP. This system is driven by the same canonical system as the motion system. In this way, both systems for the motion and tactile trajectory are synchronized and we receive the correct desired tactile signal at each time point.

To be able to differently react to the tactile information at different stages of the task execution we need varying weights of the tactile feedback that depend on the state of the canonical system. We model these weights with normalized Gaussians distributed along the state axis. The tactile feedback term is added to the forcing function of the DMP as

$$\hat{f}(z) = \frac{\sum_{i=1}^{m} \psi_i(z) w_i z}{\sum_{i=1}^{m} \psi_i(z)} + \sum_{j=1}^{n} \left(\frac{\sum_{i=1}^{k} \hat{\psi}_i(z) \hat{w}_{ij} z}{\sum_{i=1}^{k} \hat{\psi}_i(z)} (\bar{y}_j - y_j) \right),$$

where n is the length of the tactile feedback vector and $(\bar{y}_j - y_j)$ is the difference of the j-th element of the current tactile vector and the desired tactile vector, k is the number of Gaussian kernel functions for the tactile feedback weights. For a better synchronization, we use the same number of kernels as in the original forcing function and the same Gaussian functions, i.e. $m = k$ and $\hat{\psi}_i(z) = \psi_i(z)$.

C. Dimensionality reduction of tactile information

In the Section III-D, we will explain how to learn weights of the tactile coupling with reinforcement learning. However, by using each tactile element (tactel) individually, the number of weights becomes very large. For example, the tactile vector of a 8x8 tactile image has the length $n = 64$. With the number of Gaussians in the model $k = 50$ the number of weights that have to be learned for a single DMP is 3200. Therefore, we perform dimensionality reduction of tactile images. In particular, we apply principal component analysis (PCA) [7] to the tactile image vectors collected from multiple demonstrations of the same task. Consequently, we use only the largest principal components of the tactile trajectory for the feedback. Thus, only the parts of tactile images that vary throughout the task execution influence the feedback term. This significantly reduces the number of weights that have to be learned.

Further reduction of the number of tactile feedback weights can be achieved by dividing the action into phases and learning only a single weight for each phase [28]. Recognition of the action phases can be accomplished by clustering tactile images based on their similarity. Each action phase will have similar tactile images belonging to a specific cluster. As our localization is not perfect, we often do not have exact pixel-to-pixel correspondences through the image alignment. Therefore, we employ a kernel similarity measure that takes into account and tolerates these inaccuracies.

Kernel descriptors, introduced by Bo et al. [29] provide a way to compute low-level image patch features based on various pixel attributes, such as gradient, color etc. The authors use attribute and position kernels to compute the similarity between two image patches. In our work, we use the similarity of intensity values of the tactile images. The corresponding measure is computed as

$$K(P, Q) = \sum_{z \in P} \sum_{z' \in Q} k_c(c(z), c(z')) k_p(z, z'),$$

where $k_c(c(z), c(z')) = \exp(-\gamma_c \|c(z) - c(z')\|^2)$ is an intensity kernel, $k_p(z, z') = \exp(-\gamma_p \|z - z'\|^2)$ is a position kernel, P and Q are two tactile images. Gaussian kernels result in smooth transitions between similar tactile images. By changing γ_c and γ_p we can adjust the width of the Gaussians and hence, how much similarity tolerance we allow for the intensity values and pixel positions accordingly. In our experiments, we set $\gamma_c = 1$ and $\gamma_p = 2$.

Clustering of tactile images based on their kernel similarity can be performed with *spectral clustering* [6]. We apply the normalized spectral clustering algorithm by Shi and Malik [30]. We assume a fully connected similarity graph where the weight of an edge between two images X_i and X_j is their similarity value. Subsequently, clustering is performed in a lower-dimensional space based on the eigenvalues of the similarity matrix $\mathbf{S}$ where $\mathbf{S}_{ij} = K(X_i, X_j)$.

Fig. 3 shows on the left a heat-map visualization of the kernel similarity matrix of the temporarily sorted tactile images acquired while performing the scraping action. We

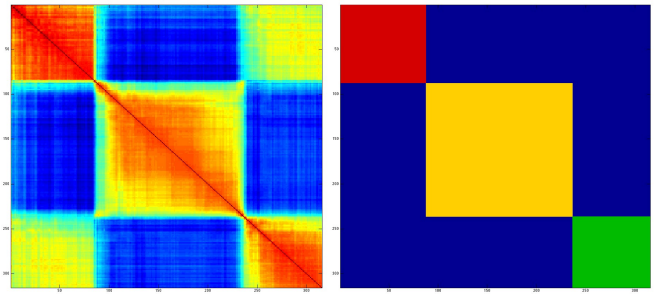

Fig. 3. Similarity matrix heat-map (left) and spectral clustering (right) of tactile images of the scraping task. *Clusters:* red (top left) - moving towards the table, yellow (center) - scraping along the table surface, green (bottom right) - moving back to the initial position

can recognize three parts of the action in the heat-map, as during these parts the tactile images are similar and therefore these portions are painted in high-temperature colors. We also notice that the first and third motion parts have high similarity values as well. They correspond to moving the hand in the air: in the first case towards the table and in the second case towards the initial position. In both cases, the tactile images are analogous. We apply spectral clustering to the similarity matrix with the number of clusters equal to three. The cluster assignments of tactile images are depicted in Fig. 3 on the right, where clusters are painted in different colors. Hence, we are able to correctly divide the action into three segments.

After applying spectral clustering, we learn only a single effective weight θ_{js} for each action phase. The weight of each Gaussian inside a phase is then computed as follows:

$$\hat{w}_{ij} = \frac{\theta_{js}}{c_i}$$

where j is the tactile vector index, s is the index of the action phase, i is the index of the Gaussian and c_i is the center of the Gaussian.

D. Policy search for learning tactile feedback weights

In this section, we explain how the tactile feedback parameters can be optimized with reinforcement learning. The main idea of reinforcement learning is to learn a controller, i.e. a policy of a robot, that maximizes a given reward. In this work, we employ policy search to learn weights of the tactile feedback controller. We define the policy $\pi(\boldsymbol{\theta})$ as a Gaussian distribution of the feedback weights with a mean $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$. At the beginning of a single task execution, i.e. an episode, the feedback weight vector $\boldsymbol{\theta}$ is sampled from this distribution. We compute the reward $R(\boldsymbol{\theta})$ by integrating the total deviation of the tactile signal from the desired tactile trajectory during the episode.

We perform optimization of the policy with episodic relative entropy policy search (REPS) [5]. The advantage of this method is that, in the process of reward maximization, the loss of information during a policy update is bounded, which leads to a better convergence behavior.

The goal of REPS is to maximize expected reward $J(\pi)$ of a policy π subject to bounded information loss. Information loss is defined as the Kullback-Leibler (KL) divergence between the old and new policies. By bounding the information loss, we limit the change of the policy and hence, avoid premature convergence.

Let $q(\boldsymbol{\theta})$ be the old policy and $\pi(\boldsymbol{\theta})$ be the new policy after the policy update. Then, we can formulate a constrained optimization problem:

$$\max_{\pi} J(\pi) = \int \pi(\boldsymbol{\theta}) R(\boldsymbol{\theta})\, d\boldsymbol{\theta} \quad \text{s. t.}$$

$$\int \pi(\boldsymbol{\theta}) \log \frac{\pi(\boldsymbol{\theta})}{q(\boldsymbol{\theta})}\, d\boldsymbol{\theta} \leq \epsilon,$$

$$\int \pi(\boldsymbol{\theta})\, d\boldsymbol{\theta} = 1,$$

where $J(\pi)$ is the total expected reward of using policy $\pi(\boldsymbol{\theta})$. The first constraint bounds the KL-divergence between the policies with the maximum information lost set to ϵ. The second constraint ensures that $\pi(\boldsymbol{\theta})$ is a proper probability distribution.

Solving the optimization problem with Lagrange multipliers results in a dual function:

$$g(\eta) = \eta\epsilon + \eta \log \int q(\boldsymbol{\theta}) \exp\left(\frac{R(\boldsymbol{\theta})}{\eta}\right) d\boldsymbol{\theta},$$

where the integral term can be approximated from samples with a maximum-likelihood estimate of the expected value. Furthermore, from the Lagrangian it can be derived that

$$\pi(\boldsymbol{\theta}) \propto q(\boldsymbol{\theta}) \exp\left(\frac{R(\boldsymbol{\theta})}{\eta}\right).$$

Therefore, we can compute the new policy parameters with a weighted maximum-likelihood solution. The weights are $\exp\left(R(\boldsymbol{\theta})/\eta\right)$, where rewards are scaled by η, which can be interpreted as the temperature of a soft-max distribution. By decreasing η we give larger weights to the high-reward samples. Increasing of η results in more uniform weights. The parameter η is computed according to the optimization constraints by solving the dual problem.

Given a set of feedback weight vectors $\{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N\}$ and corresponding episode rewards, the policy update rules for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ can be formulated as follows [31]:

$$\boldsymbol{\mu} = \frac{\sum_{i=1}^{N} d_i \boldsymbol{\theta}_i}{\sum_{i=1}^{N} d_i}, \quad \boldsymbol{\Sigma} = \frac{\sum_{i=1}^{N} d_i \left(\boldsymbol{\theta}_i - \boldsymbol{\mu}\right)\left(\boldsymbol{\theta}_i - \boldsymbol{\mu}\right)^{\top}}{\sum_{i=1}^{N} d_i},$$

with weights $d_i = \exp\left(R(\boldsymbol{\theta})/\eta\right)$

IV. Experiments

In this section, we first describe the hardware used in our robot experiments. Afterwards, we present experimental results of the in-hand object localization and tactile-based manipulation.

A. Hardware

In this work, we aimed at building a low-cost robot gripper equipped with tactile sensors. We used two dynamical matrix analog pressure sensors by *PlugAndWear.com*. One of the sensors can be seen in Fig. 2. The matrix has a sensitive area of 16cm x 16cm and contains 64 sensor cells in 8 rows and 8 columns. Hence, the size of a single cell and consequently the spatial resolution is 20mm.

We attached two tactile matrix arrays to a parallel gripper. In order to improve compliance and involve more sensor elements during an object contact we put a 5mm foam layer underneath the sensor array such that the fabric of the sensor stretches by pressing on it. Furthermore, to increase the effective resolution of sensing we placed both sensors with a 10mm shift to each other, which is a half of the size of a single tactile element.

With the given setup, the tactile data could be captured with a frequency of about 50Hz. We attached the gripper to a light-weight KUKA arm with 7 degrees of freedom. In order to be able to map single tactels to coordinates inside the robot hand, we calibrated positions of each element of the matrix array.

B. In-hand localization experiments

We evaluated the performance of the pose estimation algorithms on the tactile data acquired by performing grasps of several objects: a hammer, a screwdriver, a roll of tape, a saw and a spatula. Appearance features were extracted from intensity values of 3x3 tactile image patches.

Before collecting the tactile data, the object was placed at a known position. Subsequently, we performed 40 grasps of each object from random angles and positions. The tactile images were recorded along with the corresponding poses of the object inside the hand. We applied the same gripping force in all experiments.

The localization performance was evaluated by running leave-one-out cross-validation. That is, before performing pose inference on the hold-out grasp, we instantiated the model with the 39 other grasps. The threshold of the ICP points was chosen experimentally to discard small values of the sensor cells that are not currently in contact with the object. Furthermore, we used 10 nearest neighbors in the voting scheme experiments.

Table I shows the mean absolute errors of position and angle estimation for all methods and all objects. We could achieve the best results on leave-one-out cross-validation with the voting scheme. The main reason for that is that the other two methods required a larger number of observation in order to make reliable pose predictions. In fact, the number of tactile observations from a single grip was in the range from 11 to 32. When using PHOR, the data was distributed among the primitive features and the resulting amount of observations was too low for building reliable KDEs. The generated KDEs had much noise, which was propagated to the top of the hierarchy. The voting scheme, on contrary, used only a single KDE for all observations, which had enough data for a reliable pose inference. The convergence of the ICP method also heavily depends on the amount of observations, i.e. the number of points to be aligned. Due to the low amount of points, the number of possible alignments increases. Therefore, although we could improve the alignment with appearance features, in many cases ICP did not converge to a correct alignment and terminated in a local optimum.

	PHOR		**Voting Scheme**		**ICP**	
Object	***Pos.***	***Angle***	***Pos.***	***Angle***	***Pos.***	***Angle***
Hammer	4.09	22.54	**1.37**	**11.83**	2.19	21.89
Screwdriver	4.55	23.13	**1.56**	**10.27**	3.21	30.76
Roll of tape	3.52	16.79	**2.12**	**12.91**	2.25	26.27
Saw	3.94	22.44	**2.19**	**6.65**	2.99	26.81
Spatula	4.74	29.96	**2.96**	**22.67**	3.11	28.51
Average	4.17	22.97	**2.04**	**12.87**	2.75	26.85

TABLE I

MEAN ABSOLUTE ERRORS OF POSE ESTIMATION WITH LEAVE-ONE-OUT CROSS-VALIDATION OF 40 GRASPS. POSITION: *cm*, ANGLE: *degree*.

	PHOR		**Voting Scheme**		**ICP**	
Object	***Pos.***	***Angle***	***Pos.***	***Angle***	***Pos.***	***Angle***
Hammer	1.55	9.24	**0.66**	**9.15**	1.80	9.89
Screwdriver	1.54	**9.63**	**0.55**	10.20	2.44	14.25
Roll of tape	1.49	**6.97**	**0.60**	8.71	1.49	10.08
Saw	1.91	16.78	1.55	**2.94**	**1.51**	12.89
Spatula	2.09	17.61	**1.80**	21.69	2.35	**16.64**
Average	1.72	12.05	**0.90**	**10.54**	1.92	12.75

TABLE II

MEAN ABSOLUTE ERRORS OF POSE ESTIMATION WITH LEAVE-TEN-OUT CROSS-VALIDATION OF 40 GRASPS. POSITION: *cm*, ANGLE: *degree*.

Using more observations leads to a significant improvement of localization performance. Table II shows pose estimation results after combining tactile observations of 10 grasps for pose inference and creating models from the other 30 grasps. The increased number of grasps simulates a higher resolution of the tactile matrix. Improvement of results of PHOR and ICP methods confirms that they perform better with a bigger amount of observations. Here, all algorithms had an average positional error under 2cm, the voting scheme even under 1cm. The angular error was in the 10-13 degree range.

The lowest performance was observed for the spatula. It was the thinnest object, which led to a situation where the tactile sensors of both hands could touch each other during the grip and generate noise. The problem of detecting contacts between different parts of the robot hand, i.e. self-collisions of the robot, should be addressed in the future. Furthermore, localization of objects that are much bigger than the hand remains an open problem, as only a small object part can be observed at a time. In our experiments, a frequent kind of a localization error was a 180-degree flipping of objects with long straight handles, such as a hammer or spatula, due to the high similarity of the corresponding tactile images. This problem can be addressed by using vision to determine the coarse alignment of the object before grasping. The localization can then be refined by using the tactile data, once the object is grasped.

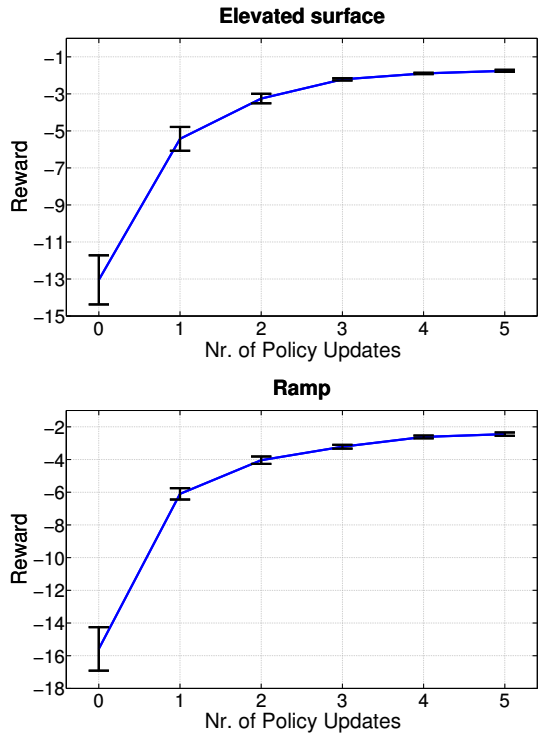

Fig. 4. Mean reward values and standard errors after each policy update. *Top:* scraping task on an elevated surface. *Bottom:* scraping task on a ramp.

Due to the low resolution of the tactile sensor, the tactile images of objects with a similar form do not show a large variation. Therefore, in many cases, we can generalize object models, e.g. for elongated or circular objects.

For the object manipulation task, we need a good localization performance with tactile data from a single grasp. As the voting scheme was the only method that could provide reliable pose estimation from a small amount of data, we used this method for our manipulation experiments.

C. Tactile-based manipulation experiments

To evaluate the proposed approach of learning manipulation skills with tactile feedback, the robot was given the task of gently scraping a surface with a spatula. After learning the movement from demonstration, we introduced two kinds of variations from the original setup. In the first task, we used a higher surface than the robot had expected by adding an elevation of 5cm. The goal of the robot was to adapt to the new height and stay close to the desired tactile trajectory by correcting the pressure of the spatula on the surface. In the second task, we placed a ramp on the table. Similarly, the goal of the robot was to learn to adjust its tactile feedback to the dynamically changing height of the surface.

In both tasks, we used the first two principal components of the desired tactile trajectory. For each principal component, we learned three weights for the three phases of the scraping action. The robot movements were encoded in a three dimensional Cartesian space with a separate DMP for each dimension. Thus, the total number of tactile feedback weights to optimize with REPS was 18. We conducted the complete policy learning experiments three times for both tasks. Each experiment included 6 sets of 20 episodes with resulting policy updates between the sets.

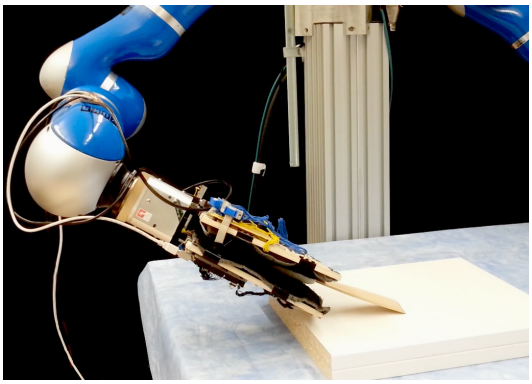

Fig. 5. Scraping task on an elevated surface before any policy updates. The robot goes too far down, resulting in a larger force and a smaller angle to the surface.

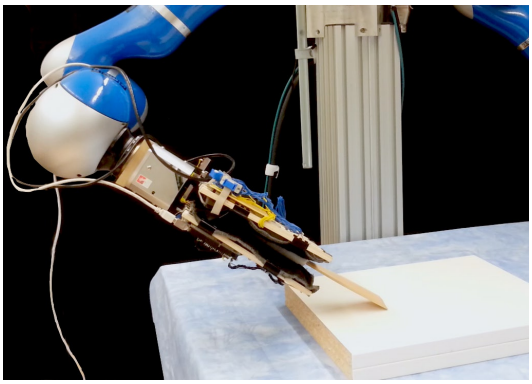

Fig. 6. Scraping task on an elevated surface after five policy updates. The robot controls the height such that the spatula gently touches the surface.

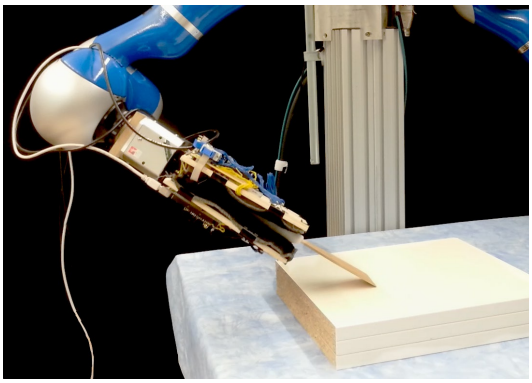

Fig. 7. Scraping task on a surface elevated to 7.5 cm and using a policy learned on a 5cm elevation. The robot still controls its height correctly.

Fig. 4 shows development of the mean reward after each policy update of all experiments on an elevated surface and a ramp. Additionally, the figures show the standard error of the reward values. Although we have some high reward episodes in the first iteration, they are not consistent. As the mean reward increases, the variance of rewards and therefore the standard error becomes smaller. This indicates, that the exploration rate decreases with each policy update and the policy converges to the tactile feedback weights with high rewards.

In the elevated surface task, the biggest weights were learned for the first phase of the task. By hitting the table sooner than expected, there was a big deviation from the desired tactile trajectory in the phase when the robot should still have been in the air, which led to a large correction. In the ramp task, substantial weights were also learned for the second action phase as the deviation from the desired signal increased during scraping along the surface of the ramp.

In both tasks, the robot could adapt to the change in the environment by learning the tactile feedback weights. Fig. 5 shows the task execution on an elevated surface before any policy updates were performed. The robot goes too far towards the surface, as it tries to reproduce the demonstration. After five policy updates, in Fig. 6 the robot learns to control its height using tactile feedback and performs the task correctly with a suitable angle and a force. By using the same policy learned on the 5cm elevation, the robot could also perform the task on the surface elevated to 7.5cm (Fig. 7). This result shows that the robot learned a policy that generalizes to different surface heights.

V. Conclusion

In this work, we explored tactile sensing for in-hand object localization and object manipulation with tactile feedback. Three pose estimation algorithms were adapted for the tactile images acquired from tactile matrix arrays. We showed that the number of tactile observations and in general the resolution of the tactile data have a big influence on their performance. The probabilistic voting scheme showed the best results for localizing objects from single grasps.

We employed dynamic motor primitives for learning a manipulation action from human demonstration and incorporated tactile feedback by computing the deviations from the desired tactile trajectory. The number of tactile feedback parameters was reduced by performing dimensionality reduction of tactile images and action phase recognition. We optimized the feedback parameters with the REPS algorithm. In real robot experiments, we showed that the tactile feedback significantly improves the movement execution in an altered task environment.

Acknowledgment

The research leading to these results has received funding from the European Community's Seventh Framework Programme under grant agreements 610878 (3rdHand) and 610967 (TACMAN).

References

[1] R. S. Johansson and R. J. Flanagan, "Coding and use of tactile signals from the fingertips in object manipulation tasks.," *Nature Reviews Neuroscience*, vol. 10, pp. 345–359, April 2009.

[2] S. Chitta, J. Sturm, M. Piccoli, and W. Burgard, "Tactile sensing for mobile manipulation.," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 558–568, 2011.

[3] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots.," in *ICRA*, pp. 1398–1403, IEEE, 2002.

[4] J. Kober, B. Mohler, and J. Peters, "Learning perceptual coupling for motor primitives.," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pp. 834–839, 2008.

[5] J. Peters, K. Mülling, and Y. Altun, "Relative entropy policy search.," in *AAAI*, AAAI Press, 2010.

[6] U. Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[7] I. T. Jolliffe, *Principal component analysis*. New York: Springer, 1986.

[8] O. Kroemer, C. H. Lampert, and J. Peters, "Learning dynamic tactile sensing with robust vision-based training.," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 545–557, 2011.

[9] A. Schneider, J. Sturm, C. Stachniss, M. Reisert, H. Burkhardt, and W. Burgard, "Object identification with tactile sensors using bag-of-features.," in *IROS*, pp. 243–248, IEEE, 2009.

[10] A. Petrovskaya, O. Khatib, S. Thrun, and A. Y. Ng, "Touch based perception for object manipulation.," *Robotics Science and Systems, Robot Manipulation Workshop*, 2007.

[11] C. Corcoran and R. Platt, "A measurement model for tracking hand-object state during dexterous manipulation.," in *ICRA*, pp. 4302–4308, IEEE, 2010.

[12] R. Li, R. Platt, W. Yuan, A. ten Pas, N. Roscup, and E. Adelson, "Localization and manipulation of small parts using gelsight tactile sensing," in *IROS*, IEEE, 2014.

[13] K. Hsiao, L. P. Kaelbling, and T. Lozano-Prez, "Task-driven tactile exploration.," in *Robotics: Science and Systems* (Y. Matsuoka, H. F. Durrant-Whyte, and J. Neira, eds.), The MIT Press, 2010.

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

[15] H. Dang and P. Allen, "Stable grasping under pose uncertainty using tactile feedback," *Autonomous Robots*, vol. 36, no. 4, pp. 309–330, 2014.

[16] K. Muelling, J. Kober, O. Kroemer, and J. Peters, "Learning to select and generalize striking movements in robot table tennis," *International Journal of Robotics Research*, no. 3, pp. 263–279, 2013.

[17] J. Peters and S. Schaal, "Policy gradient methods for robotics.," in *IROS*, pp. 2219–2225, IEEE, 2006.

[18] O. Kroemer, R. Detry, J. H. Piater, and J. Peters, "Combining active learning and reactive control for robot grasping.," *Robotics and Autonomous Systems*, vol. 58, no. 9, pp. 1105–1116, 2010.

[19] P. Englert, A. Paraschos, M. P. Deisenroth, and J. Peters, "Probabilistic model-based imitation learning.," *Adaptive Behaviour*, vol. 21, no. 5, pp. 388–403, 2013.

[20] P. Pastor, M. Kalakrishnan, L. Righetti, and S. Schaal, "Towards associative skill memories.," in *Humanoids*, pp. 309–315, IEEE, 2012.

[21] R. Detry, N. Pugeault, and J. Piater, "A probabilistic framework for 3D visual object representation.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 10, pp. 1790–1803, 2009.

[22] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman and Hall/CRC, April 1986.

[23] E. B. Sudderth, A. T. Ihler, M. Isard, W. T. Freeman, and A. S. Willsky, "Nonparametric belief propagation.," *Communications of the ACM*, vol. 53, no. 10, pp. 95–103, 2010.

[24] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images.," *Image and Vision Computing*, vol. 10, no. 3, pp. 145 – 155, 1992.

[25] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures.," *Commun. ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[26] D. Glasner, M. Galun, S. Alpert, R. Basri, and G. Shakhnarovich, "Viewpoint-aware object detection and pose estimation.," in *ICCV*, pp. 1275–1282, IEEE, 2011.

[27] Y. Cheng, "Mean shift, mode seeking, and clustering.," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 17, no. 8, pp. 790–799, 1995.

[28] O. Kroemer, H. van Hoof, G. Neumann, and J. Peters, "Learning to predict phases of manipulation tasks as hidden states," in *Proceedings of 2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[29] L. Bo, X. Ren, and D. Fox, "Kernel descriptors for visual recognition.," in *NIPS*, pp. 244–252, Curran Associates, Inc., 2010.

[30] J. Shi and J. Malik, "Normalized cuts and image segmentation.," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, 2000.

[31] M. P. Deisenroth, G. Neumann, and J. Peters, "A survey on policy search for robotics.," *Foundations and Trends in Robotics*, vol. 2, no. 1-2, pp. 1–142, 2013.