Object Manifold Learning with Action Features for Active Tactile Object Recognition

Daisuke Tanaka, Takamitsu Matsubara, Kentaro Ichien, and Kenji Sugimoto

***Abstract*— In this paper, we consider an object recognition problem based on tactile information using a robot hand. The robot performs an exploratory action to the object to obtain the tactile information, however, poorly designed actions may not be sufficiently informative. In contrast, if we could collect sample data by sequentially performing informative actions, i.e., *active learning*, the required time would be drastically reduced. To this end, we propose a novel approach for active tactile object recognition. Our approach combines both an active learning scheme and a nonlinear dimensionality reduction method. We first extracts the *object manifold*, each coordinate of which represents an object, from tactile sensor data and action features using Gaussian Process Latent Variable Models. At the same time, a probabilistic model of the observed data related to the action and the object are learned. Then, with the learned model, optimally-informative exploratory actions can be computed sequentially, and performed to efficiently collect the data for recognition. We show experimental results that verify the effectiveness of our proposed method with synthetic data and a real robot.**

I. INTRODUCTION

Object recognition using a robot hand based on tactile information such as pressure, vibration and temperature is a crucial problem (Fig. 1). To recognize an object by such a robot, the following procedures are executed: (1) an exploratory action is designed, (2) the robot performs the action to the object, (3) the recognition task is accomplished with the obtained tactile information.

The most important procedure may be (1): Poorly designed (e.g, random or hand-tuned) actions may not be sufficiently informative, and would require unreasonable amount of time to accomplish the task. In contrast, if we could collect sample data by sequentially performing informative actions, i.e., *active learning*, the required time would be drastically reduced. The effectiveness of active learning has been investigated in (e.g. [1]–[7]). See the discussion section for the details.

Active learning requires the *observation model* that relates the observed data to the action and the object (state) to seek informative actions. We consider to learn such a model from training data using Gaussian Processes (GPs) [8], that relates the observed tactile sensor data to the continuous object and action parameters, to enjoy the compatibility of GPs with the active learning based on mutual information, as well as in [1]. However, in the object recognition task, the suitable representations of the objects for the object parameters in the model are not given a priori and they might be strongly task-dependent, unlike in [1]. Besides, using unsuitable object parameters may deteriorate the task performance.

All authors are with the Graduate School of Information Science, Nara Institute of Science and Technology, Japan, {daisuke-t, takam-m, kentaro-ic, kenji}@is.naist.jp

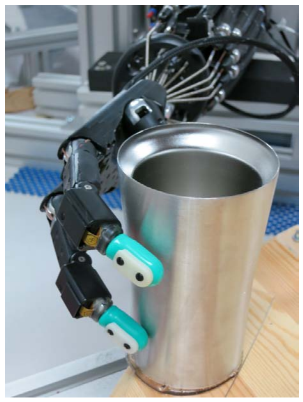


Fig. 1. Tactile object recognition by a robot hand.

To cope with the problem, we propose a novel approach for efficient tactile object recognition using active learning. Our approach combines both an active learning scheme and a nonlinear dimensionality reduction method. We first extract the *object manifold* using Gaussian Process Latent Variable Models (GPLVMs) [9], that embeds the tactile sensor data observed by touching different objects in the observation space into a lower dimensional space (each coordinate corresponds to an object parameter) with the observed data and action parameters. At the same time, a probabilistic model of the observed data related to the action and object parameters are learned. Then, with the learned model, optimally-informative exploratory actions can be sequentially computed, and performed to efficiently collect the data for object recognition, by following [1]. Therefore, our method can be regarded as generalization of the framework [1] to be suitable for the object recognition task.

Alternatively, the differential geometry based feature [10] and the force-distance profiles based feature [11] for tactile sensor data could be used as the object parameters. Since these features are strongly related to the physical quantities such as a surface shape and hardness of the object, the suitable features need to be selected for the task a priori. Data-driven feature extraction methods based on dimensionality reduction methods such as Principal Component Analysis [12], Self-Organizing Maps [13], [14], and Maximum Covariance Analysis [15] have also been explored; however, these methods are limited in a single action and unclear how to use for computing informative actions with the notable exception of [5].

II. PROPOSED METHOD

In this section, we present our proposed method for active tactile object recognition, which is composed of two parts: the modeling method with object manifold learning, and the active object recognition method with the learned model.

A. Object Manifold Learning with Action Features

On the object recognition problem, the object parameters are defined for individual objects and extracted from the tactile data obtained by actions. If different objects are touched by the same action, different tactile data are observed. Also, when different actions are performed to the same objects, different tactile data are observed. Therefore, the object parameters need to be extracted together with both the tactile data and the action parameters. We present a method that simultaneously achieves both object parameter extraction and model learning using GPLVMs.

1) Tactile Data Modeling using GPs: We now start to model the relationship among tactile data, object and action. The following nonlinear function $\mathbf{f}$ of the observation $\mathbf{y} \in \mathbb{R}^{d_{\mathbf{y}}}$ is assumed:

$$\mathbf{y} = \mathbf{f}(\mathbf{s}, \mathbf{x}) + \boldsymbol{\epsilon}, \tag{1}$$

where the first variable $\mathbf{s} \in \mathbb{R}^{d_{\mathbf{s}}}$ is the object parameter of the touched object, and the second variable $\mathbf{x} \in \mathbb{R}^{d_{\mathbf{x}}}$ is the action parameter which represents the exploratory action used for obtaining the observation (tactile data). In our problem setting, only $\mathbf{y}$ and $\mathbf{x}$ are assumed to be known as training data. Moreover, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\epsilon})$ is observation noise distributed normally with zero mean and covariance $\boldsymbol{\Sigma}_{\epsilon} = \mathrm{diag}\left\{\sigma_1^2, \sigma_2^2, \ldots, \sigma_{d_{\mathbf{y}}}^2\right\}$.

The nonlinear function $\mathbf{f}$ is modeled using GPs. We consider the GP model for each dimension of $\mathbf{y}$ as follows:

$$y_a \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_a), \tag{2}$$

for $a = 1, 2, \ldots, d_{\mathbf{y}}$. Here, observation noise turns into $\epsilon_a \sim \mathcal{N}(0, \sigma_a^2)$, and training data set is $\mathcal{D}_a = \{\mathcal{D}_{a,1}, \ldots, \mathcal{D}_{a,L}\}$, $\mathcal{D}_{a,\ell} = \left\{\mathbf{x}^{(\ell,i)}, y_a^{(\ell,i)}\right\}_{i=1}^{N_\ell}$, where L is the number of recognition target objects. In this GP model, it is assumed that $\mathbf{s}$ and $\mathbf{x}$ are independent for model simplification, and the following squared exponential kernel function is considered:

$$k_a(\mathbf{z}, \mathbf{z}') = \alpha_a^2 \exp\left(-\frac{1}{2}(\mathbf{z} - \mathbf{z}')^{\mathrm{T}} \mathbf{H}_a^{-1} (\mathbf{z} - \mathbf{z}')\right).$$

$\mathbf{z} = \left[\mathbf{s}^{\mathrm{T}}, \mathbf{x}^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^{d_{\mathbf{z}}}$ is defined for simple representation, where $d_{\mathbf{z}} = d_{\mathbf{s}} + d_{\mathbf{x}}$. α_a^2 is the variance of f_a. $\mathbf{H}_a = \text{block diag}\left\{\mathbf{H}_a^{\mathbf{s}}, \mathbf{H}_a^{\mathbf{x}}\right\}$, where $\mathbf{H}_a^{\mathbf{s}} \in \mathbb{R}^{d_{\mathbf{s}} \times d_{\mathbf{s}}}$ and $\mathbf{H}_a^{\mathbf{x}} \in \mathbb{R}^{d_{\mathbf{x}} \times d_{\mathbf{x}}}$ are diagonal matrices with positive elements, adjusts the scale of each dimension of $\mathbf{s}$ and $\mathbf{x}$ respectively. Note that this model is one example of Multifactor Gaussian Process Models [16].

2) Object Parameter Extraction by Manifold Learning: We consider to extract the object parameter set $\mathcal{S} = \{\mathbf{s}^{(1)}, \mathbf{s}^{(2)}, \ldots, \mathbf{s}^{(L)}\}$ using the GP model (2) with a similar way to the GPLVMs. The GPLVMs assume an observation $\mathbf{y}$ generated from a low dimensional variable $\boldsymbol{\xi}$ through a nonlinear function $\mathbf{g}$ with additional observation noise $\boldsymbol{\eta}$, $\mathbf{y} = \mathbf{g}(\boldsymbol{\xi}) + \boldsymbol{\eta}$. The solution to $\mathbf{x}$ is found together with the hyperparameters by maximizing log-likelihood of its GP model w.r.t. $\mathbf{x}$ and the hyperparameters. Please refer to [9] for the details. We apply the same optimization technique to our problem setting. For our model (2), the log-likelihood function is defined as,

$$\begin{aligned}&\log p(\mathbf{y}_a | \mathcal{S}, \mathbf{X}, \gamma_a) \\ &\quad = -\frac{1}{2}\log\det\mathbf{K}_a - \frac{1}{2}\mathbf{y}_a^{\mathrm{T}}\mathbf{K}_a^{-1}\mathbf{y}_a - \frac{1}{2}\log(2\pi),\end{aligned} \tag{3}$$

where $\mathbf{K}_a \in \mathbb{R}^{N \times N}$ is the gram matrix with $K_{a,ij} = k_a(\mathbf{z}^{(i)}, \mathbf{z}^{(j)}) + \sigma_a^2 \delta_{ij}$ as (i,j)th-element, and $\mathbf{z}^{(i)}$ corresponds to the i-th column of the matrix $\mathbf{Z} \in \mathbb{R}^{d_{\mathbf{z}} \times N}$ defined as,

$$\mathbf{Z} = \begin{bmatrix} \mathbf{s}^{(1)} & \cdots & \mathbf{s}^{(1)} & \cdots & \mathbf{s}^{(L)} & \cdots & \mathbf{s}^{(L)} \\ \mathbf{x}^{(1,1)} & \cdots & \mathbf{x}^{(1,N_1)} & \cdots & \mathbf{x}^{(L,1)} & \cdots & \mathbf{x}^{(L,N_L)} \end{bmatrix}, \tag{4}$$

where $N = \sum_{\ell=1}^{L} N_\ell$. $\mathbf{X}$ is a $d_{\mathbf{x}} \times N$ sub-matrix of $\mathbf{Z}$, which is a part of $\mathbf{x}$ in $\mathbf{Z}$, and $\mathbf{y}_a \in \mathbb{R}^N$ is defined as,

$$\mathbf{y}_a = \begin{bmatrix} y_a^{(1,1)} & \cdots & y_a^{(1,N_1)} & \cdots & y_a^{(L,1)} & \cdots & y_a^{(L,N_L)} \end{bmatrix}^{\mathrm{T}},$$

and $\gamma_a = \left\{\mathbf{H}_a^{\mathbf{s}}, \mathbf{H}_a^{\mathbf{x}}, \alpha_a^2, \sigma_a^2\right\}$ is a hyperparameter set. Note that $\mathbf{s}$ is the only unknown variable in the latent variable $\mathbf{z}$, and $\mathbf{s}^{(\ell)}$ is N_ℓ times included in $\mathbf{Z}$. With holding this structure, $\mathcal{S}$ and $\gamma = \{\gamma_1, \ldots, \gamma_{d_{\mathbf{y}}}\}$ are optimized simultaneously by maximizing the log-likelihood function for all dimensions of $\mathbf{y}$ as

$$(\mathcal{S}^*, \gamma^*) \leftarrow \underset{\mathcal{S},\gamma}{\arg\max} \sum_{a=1}^{d_{\mathbf{y}}} \log p(\mathbf{y}_a | \mathcal{S}, \mathbf{X}, \gamma_a). \tag{5}$$

3) Relationship with Other Methods: Let us discuss the relationship with other methods, Saal et al. [1] and GPLVM [9], by comparing those problem settings. The differences of the problem settings are summarized in Table I. From the point of view of model learning, the true-object parameter $\boldsymbol{\theta}$ is known as training data in Saal et al [1], however, its equivalent parameter $\mathbf{s}$ is unknown in our problem setting. Next, our model (1) seems to be the same model as considered in GPLVM with defining one latent variable $\mathbf{z}$, however, $\mathbf{x}$ which is a part of latent variable $\mathbf{z}$ is known and included in training data. In addition, we have N training data, however, the number of the recognition target objects is L, and generally $L \ll N$. Accordingly, the structure of latent variables corresponding to the observation variable should be fixed as shown in Eq. (4).

B. Active Tactile Object Recognition Method

Using obtained GP model, we construct the active tactile object recognition method, by following [1]. The overview of the whole process is shown in Fig. 2.

The active tactile object recognition will be achieved as follows: First, the object's belief is initialized as a probability distribution $p_0(\mathbf{s}) = \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$. Next, the exploratory action defined by the action parameter $\mathbf{x}$ is determined by maximizing mutual information, and it is executed for the target object. Object's belief $p(\mathbf{s})$ is sequentially updated

TABLE I

DIFFERENCES OF THE PROBLEM SETTINGS AMONG SAAL ET AL. [1], OURS, AND GPLVM [9]

	Saal et al. [1] (GPs)	ours	GPLVMs [9]
model	multifactor, $\mathbf{y} = \mathbf{f}(\boldsymbol{\theta}, \mathbf{x})$	multifactor, $\mathbf{y} = \mathbf{f}(\mathbf{s}, \mathbf{x})$	singlefactor, $\mathbf{y} = \mathbf{g}(\boldsymbol{\xi})$
training data set	$\{\mathbf{y}^{(i)}, \boldsymbol{\theta}^{(i)}, \mathbf{x}^{(i)}\}$	$\{\mathbf{y}^{(i)}, \mathbf{x}^{(i)}\}$	$\{\mathbf{y}^{(i)}\}$
latent variables	—	$\mathbf{s}$	$\boldsymbol{\xi}$
number of latent variables	—	same as target object	same as training data

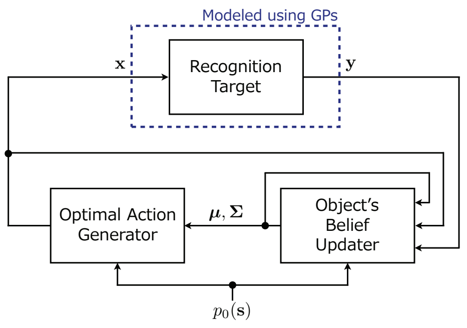


Fig. 2. Overview of our active object recognition system. We first set the probability distribution of the object parameter $p_0(\mathbf{s}) = \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ as initial object's belief. Based on the belief, the action $\mathbf{x}$ is computed and performed for the recognition target and accordingly the observation $\mathbf{y}$ is obtained. Object's belief $p(\mathbf{s})$ is updated using $\mathbf{x}$ and the obtained $\mathbf{y}$.

using the observation $\mathbf{y}$ and the action parameter $\mathbf{x}$. Based on updated $p(\mathbf{s})$, the next exploratory action $\mathbf{x}$ is determined. This procedure is repeated until T times updated or terminated if the update of mean is sufficiently small, and then the recognition task is finally achieved by the nearest-neighbor object on the extracted object manifold. We simply describe below the optimal action generation method and object's belief update law. See [1] for the details.

1) Optimal Action Generation Method: We consider the probability distribution as object's belief at the t-th update, $p_t(\mathbf{s}) = \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$, for $t = 0, 1, \ldots$. We first set the initial belief $p_0(\mathbf{s}) = \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$.

Using $p_{t-1}(\mathbf{s})$, optimal exploratory action parameter at time t, $\mathbf{x}_t$, is determined by maximizing the mutual information between $\mathbf{s}$ and $\mathbf{y}$ defined by $\boldsymbol{\mu}_{t-1}$, $\boldsymbol{\Sigma}_{t-1}$, and $\mathbf{x}$,

$$\mathbf{x}_t \leftarrow \arg\max_{\mathbf{x}} U_1(\mathbf{x}, \boldsymbol{\mu}_{t-1}, \boldsymbol{\Sigma}_{t-1}),$$

where U_1 is a function corresponding to the mutual information (See appendix for the details).

2) Object's Belief Update Method: When the observation $\mathbf{y}_t$ is obtained, belief $p_t(\mathbf{s})$ is updated by,

$$(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) \leftarrow U_2(\boldsymbol{\mu}_{t-1}, \boldsymbol{\Sigma}_{t-1}, \mathbf{y}_t, \mathbf{x}_t).$$

The function U_2 is based on Bayes' rule, and this is a Extended Kalman Filter like update, which means the distribution of $\mathbf{y}$ is locally linearized at $(\mathbf{s}, \mathbf{x}) = (\mathbf{s}_{t-1}, \mathbf{x}_t)$ (See appendix for the details). Although the Monte Carlo sampling-based updating method is shown in [1], we use the analytical Gaussian updating method for its simplicity.

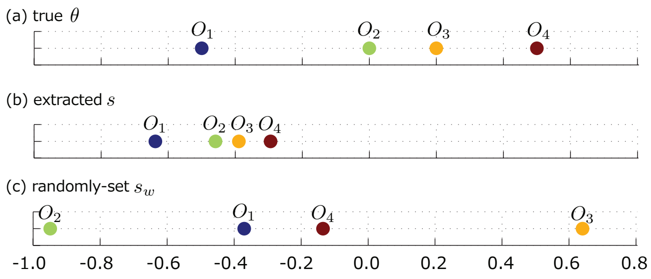


Fig. 3. Comparison along (a) true-object parameter θ, (b) extracted object parameter s, and (c) randomly-set object parameter s_w.

III. EXPERIMENTS

A. Experiment 1: Synthetic Data

1) Setting: We considered the following nonlinear function,

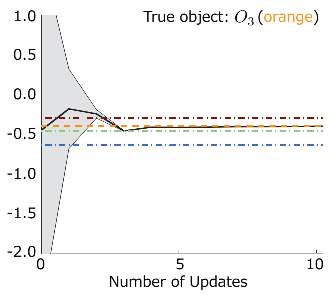
$$\mathbf{y} = \mathbf{h}(\theta, x) + \boldsymbol{\epsilon}, \tag{6}$$
$$\mathbf{h}(\theta, x) = \begin{bmatrix} \exp\left(-(x-\theta)^2\right) \\ \exp\left(-(x-\theta^2)^2\right) \\ \exp\left(-(x-\theta^3)^2\right) \end{bmatrix},$$
$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathrm{diag}\,\{0.1, 0.1, 0.1\}),$$

where $\mathbf{y} \in \mathbb{R}^3$ is the observation, $\theta \in \mathbb{R}$ is the true-object parameter (which will be the target of the estimation of object parameter s), and $x \in \mathbb{R}$ is the action parameter.

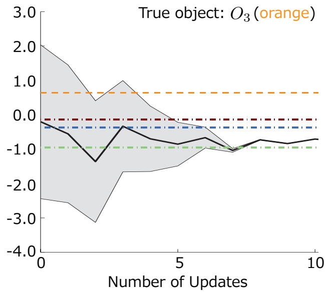
By setting $\theta = -0.5, 0, 0.2, 0.5$, and x as following normal distribution $\mathcal{N}(0, 1)$, we generated $N_\ell = 100$ training samples for each θ using Eq. 6. Therefore, the total number of training samples N is 400.

The object parameter set $\mathcal{S}$ was extracted and the GP model was simultaneously learned under the following condition: the dimension of the object parameter was set as $d_\mathbf{s} = 1$ manually, and initial values of $\mathcal{S}$ were randomly chosen, and initial hyperparameters γ_a for all a were manually selected and then optimized numerically. In the recognition task, the belief update was executed for $T = 10$. The optimal action parameter $\mathbf{x}$ on each update was determined by numerically maximizing the mutual information.

2) Result of Object Manifold Learning: The extracted object manifold from the training data is shown in Fig. 3. The parameter θ stands for the true-object parameter, and the parameter s indicates the extracted object parameters by our method. Since the manifold learning has an ambiguity of the extracted object parameters for its scaling and shifting, we verify the accuracy by the correlation coefficient between θ and s, and the value was 0.9995. Thus, the effectiveness of our object parameter extraction method was confirmed.



(a) With extracted object parameter s



(b) With randomly-set object parameter s_w

Fig. 4. Recognition result with the extracted and the randomly-set object parameter s and s_w respectively shown in Fig. 3. In this simulation, the true object is set to O_3 and its corresponding object parameter is drawn using orange dash-dot line.

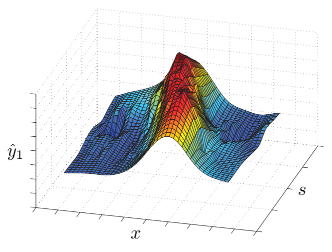
3) Result of Active Object Recognition Simulation: To verify the suitability of the extracted manifold and learned model for active object recognition, we compared its performance of the proposed method for the active object recognition task to that with a GP model using randomly-set object parameter s_w. The hyperparameters of this model is optimized in the same way as the model with s. The correlation coefficient between θ and s_w was 0.3449. The mean and covariance of initial belief were set as $\mu_0 = \bar{s}$, where $\bar{s}$ is the mean of $\mathcal{S}$, and $\Sigma = 5$ respectively.

The recognition results are shown in Fig. 4. In this recognition simulation, the true object was O_3 represented by the orange dashed line in the figure. As the result, in the proposed method the estimated object parameter s successfully and quickly converged to the true value, however, in the comparison the estimated parameter s_w was slower and converged to a wrong value at the end. Let us investigate why this difference occurred by comparing the learned two models. The both models for the 1st dimension of the observation $\hat{y}_1$ are shown in Fig. 5. In this figure, the vertical axis ($\hat{y}_1$) shows the mean of the prediction distribution for the pair (s, x). As you can see, the model with randomly-set object parameters does not have much smoothness in terms of the coordinate x. This would make updating the parameters in active learning difficult since its update law described in Section 3 involves local linearization.

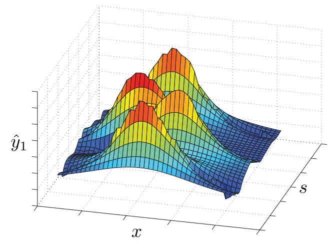
Consequently, the suitability of our proposed object's belief update method and model learning method for active object recognition was verified with the synthetic data.

B. Experiment 2: Active Tactile Object Recognition

1) Setting: We prepared $L = 4$ objects as recognition targets shown in Fig. 6(a). This experiment was done with the robot hand (Shadow Dexterous Hand by Shadow Robot Company), and the tactile sensor mounted on its fingertip (BioTac by SynTouch) shown in Fig. 6(b). The whole flow of the experiment is shown in Fig. 7. While this robot hand has 12 DoFs, in this experiment we focused on 2 DoF, FFJ3 and FFJ4, as shown in Fig. 6(b). These joints can generate actions that correspond to inflective and horizontal movements of the index finger, respectively. This robot hand with the

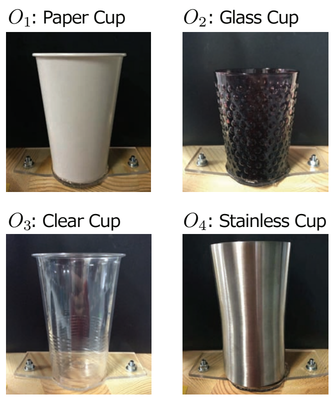


(a) With extracted object parameter $\mathbf{s}$

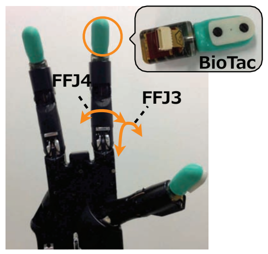


(b) With randomly-set object parameter $\mathbf{s}_w$

Fig. 5. Comparison of the GP model of $\hat{y}_1$ which is the 1st dimension of observation.



(a)



(b)

Fig. 6. Settings for Experiment 2. (a) Recognition target objects. O_1: disposable paper cup. O_2: bumpy glass cup. O_3: disposable clear cup. O_4: stainless cup. (b) 12DoF robot hand and tactile sensor. We use 2 DoF, FFJ3 and FFJ4 corresponding to inflective and horizontal movements of the index finger, respectively. On its fingertips, the BioTac sensor is mounted to obtain the tactile information.

sensor is controlled using Robot Operating System (ROS) [17]. Its control rate and sensor data collection frequency were both 1000 [Hz]. We developed the automated object switching system (1DoF) as shown in Fig. 8, that can mount 10 objects at maximum and its angular resolution is 0.2 [deg]. Therefore, the system allowed to automatically collect training data for different objects.

We describe below the details of the action parameter, tactile sensor data, and the setting of model learning and recognition simulation.

Action Parameters: We defined the action parameter based on Dynamic Movement Primitives (DMPs). DMPs represent the trajectory $\mathcal{Y}$ using nonlinear function e.g. $\mathcal{Y} = F(\mathbf{w}, \mathcal{P})$, where $\mathbf{w}$ is the parameter which defines the shape of trajectory, and $\mathcal{P}$ is the hyperparameter set containing time constants, goal state, basis functions, and so on. Please refer to [18] for the details.

In this experiment, we first obtained the basic parameter $\mathbf{w}_{\text{teach}}$ from a teaching trajectory, and generated the new trajectory using the new parameter $\mathbf{w} = \mathbf{w}_{\text{teach}} + \mathbf{x}$, where $\mathbf{x}$ is set as the action parameter. The detail is as follows: by using a cyberglove as the master system, we controlled the robot hand as the slave system so that the fingertip pushed into the object, and then started to slide the object subsequently using FFJ3 and FFJ4. The duration of each action is around 14.4 [sec]. Then, the recorded robot trajectories were approximated by two DMPs with 25 basis functions, and each basic parameter $\tilde{\mathbf{w}}^{\mathrm{FFJ3}}_{\mathrm{teach}} \in \mathbb{R}^{25}$ and $\tilde{\mathbf{w}}^{\mathrm{FFJ4}}_{\mathrm{teach}} \in \mathbb{R}^{25}$ were learned using a least square method. To reduce the dimension of the action parameter, we selected three dominant parameters out of 25 parameters, $\mathbf{w}^{\mathrm{FFJ3}}_{\mathrm{teach}} \in \mathbb{R}^{3}$ and $\mathbf{w}^{\mathrm{FFJ4}}_{\mathrm{teach}} \in \mathbb{R}^{3}$, respectively. Finally, we obtained $\mathbf{w}_{\mathrm{teach}} = [(\mathbf{w}^{\mathrm{FFJ3}}_{\mathrm{teach}})^{\mathrm{T}}, (\mathbf{w}^{\mathrm{FFJ4}}_{\mathrm{teach}})^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{6}$. Therefore, the dimension of the action parameter turned into $d_{\mathbf{x}} = 6$.

Tactile Information: BioTac sensor gives pressure, vibration, and temperature as tactile information. In this experiment, $d_{\mathbf{y}} = 5$ dimensional tactile feature was used; vibrations (2 dimensional)[1], pressure, heat flux, and temperature, all of which were obtained by using ROS. Since one action parameter corresponds to one trajectory (time series), we defined $\mathbf{y}$ as the mean of the time series of tactile sensor data and used for the following experiments.

Model Learning and Recognition Simulation: We collected $N_\ell = 100$ training data from each target object for constructing a model using the actual robot hand. Using $N = 400$ training data in total, the object parameter set $\mathcal{S}$ were extracted and the GP model was simultaneously learned with $d_{\mathbf{s}} = 2$ which was manually selected. In this model learning, a differential evolution scheme [19] was used since the marginal likelihood function has local maxima, and this problem is more serious as compared to Experiment 1 that has less parameters to be optimized. Other settings in model learning were the same as in Experiment 1. To execute the recognition task, the observation $\mathbf{y}$ was sampled from the constructed GP model, and the mean and covariance of initial belief were set to $\boldsymbol{\mu}_0 = \bar{\mathbf{s}}$ where $\bar{\mathbf{s}}$ is the mean of $\mathcal{S}$, and $\boldsymbol{\Sigma} = \mathrm{diag}\{5, 5\}$, respectively. The belief update was executed for $T = 15$. The optimal action parameter $\mathbf{x}$ on each update is set by numerical maximization.

2) Result of Object Manifold Learning: Fig. 9 shows the extracted object manifold.

Let us discuss the result; O_4 (stainless cup) is located far from other three objects since its heat characteristic is particularly different. O_1 (paper cup) and O_3 are located nearly based on their similar hardness. The appropriateness of these placement will be thoroughly validated with more objects in the near future.

3) Result of Active Object Recognition Simulation: The recognition simulation result and the trajectories corresponding to the computed action parameters at each update are shown in Figs. 10 and 11 respectively. In Fig. 10 the estimation result of both active learning and passive learning are shown and first 5 trajectories are also shown in Fig. 11 because of limited space. By using active learning, the recognition is successfully achieved since the mean of $p(\mathbf{s})$ is converging to the true object O_1 as you can see in Fig. 10(a). Comparing the result of active learning, passive

[1]Note that this tactile sensor has only 1 vibration sensor, however, the measurement frequency of vibration data is 2000 [Hz], while the collection frequency is 1000 [Hz]. The 2nd dimension of the vibration data contains 0.5 [ms] late behind the 1st data.

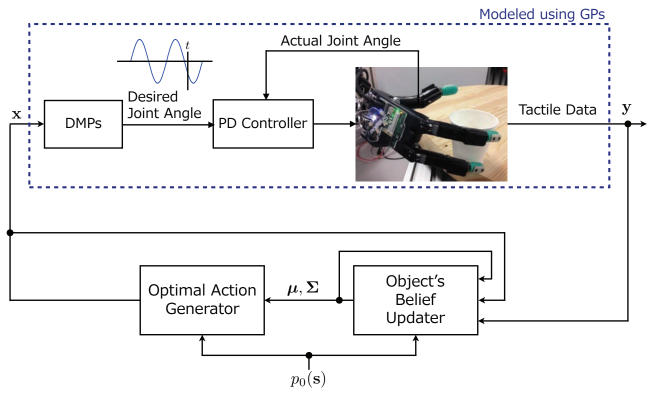


Fig. 7. Overview of recognition experiment with the real robot hand shown in Fig. 6(b). The action parameter $\mathbf{x}$ is converted to a trajectory via DMPs. The robot follows the desired joint angle in the converted trajectory using a PD controller. Consequently, the tactile data $\mathbf{y}$ is obtained. Remaining parts are same as described in Fig. 2.

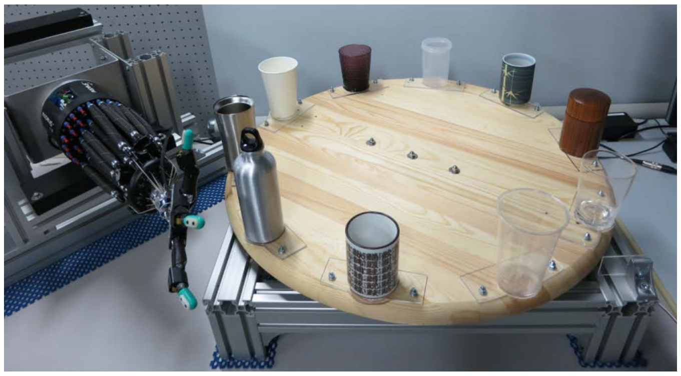
learning (Fig. 10(b)) does not converge to the true object parameter. We consider this is because of high dimensionality of action space. Fig. 11 shows the generated trajectories of exploratory actions on each update. Regarding the result of active learning, in the first update, the movement of FFJ4 is a bit stronger and of FFJ3 is a bit weaker as compared to the teaching trajectory. The difference between the generated movement and the teaching trajectory, however, this movement drastically reduces the variance of $\hat{s}_2$ (related to the uncertainty of the estimation). The following movements gradually reduce the variance of $\hat{s}_1$. The movements are larger in passive learning as opposed to active learning, however, these movements did not make the variance small. As a result, active learning generates the proper movements to obtain the most informative observation.

From these results, it was confirmed that the model constructed by our proposed method works well for the active object recognition problem even with a real robot data.

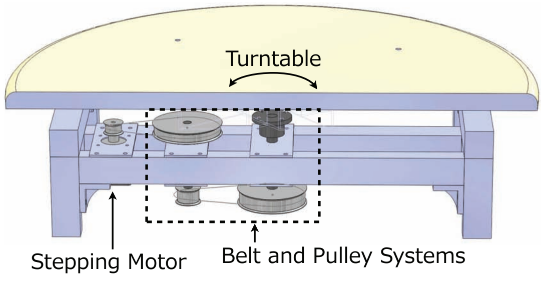
IV. DISCUSSIONS

We tackled an object recognition problem with active learning, and we proposed the object manifold learning method in order to construct a model for the action optimality evaluation. Our contributions are as follows: (1) a data-driven approach for obtaining the object parameters is proposed, i.e. object manifold learning with action features. (2) With the object manifold learning, generalization of the GP based active learning method proposed in [1] is achieved for object recognition problems. The effectiveness of our proposed method was verified through experiments with synthetic and real robot data.

Let us discuss several directions for future work. We validated our method using 4 actual objects, We are now conducting the experiments with more objects to validate scalability of our method. In addition, continuous actions are considered in our method in contrast with using discrete actions [2], [3], [6], [7] (e.g. grasp, shake). Comparison between these methods and our method would be interesting for future work. An extension of scalability of the method



(a) Overview



(b) Cross section by its CAD model

Fig. 8. Object switching system for obtaining training data. The turntable (wooden round table) turns by the stepping motor and it is controlled using the same computer for the robot hand. The torque from the stepping motor is transferred to the turntable through a belt and pulley systems to amplify it.

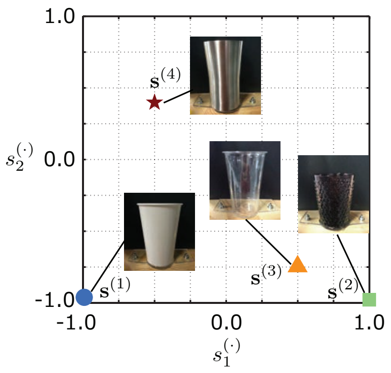
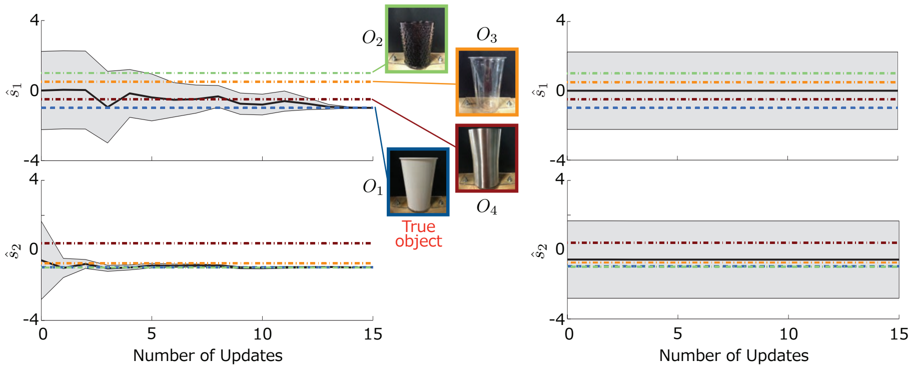


Fig. 9. Extracted object parameters of 4 objects.



(a) with active learning

(b) with passive learning

Fig. 10. Transitions of mean and standard deviation of object's belief at t-th update, $p_t(\mathbf{s}) = \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ during learning. True object in this simulation is O_1 (paper cup, represented using blue dash-dot line). The black line represents the mean of $\boldsymbol{\mu}_t$, and the gray area around the black line corresponds to the standard deviations (the square root of diagonal elements of $\boldsymbol{\Sigma}_t$).

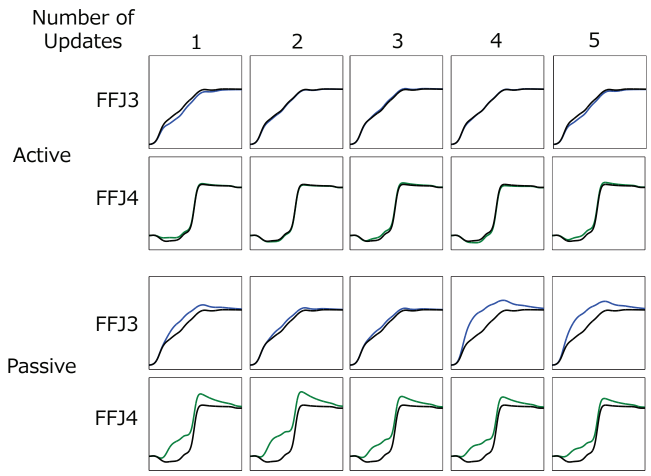


Fig. 11. Generated trajectories of exploratory actions on each update in recognition. Blue and green lines represent the trajectory generated using $\mathbf{w}_{\text{teach}} + \mathbf{x}_t$, and black lines represent the trajectory generated using $\mathbf{w}_{\text{teach}}$. Time is on the x-axis and the joint angles on the y-axis. The movements in passive learning are larger as opposed to active learning, however, these movements are not informative.

is also our future work. The high dimensional action space could make the model learning and the optimal action search intractable since a huge number of training data are required. Applying a concept of muscle synergies [20] or other dimensionality reduction scheme [21] for the action space can be considered. Moreover, we may use the information from two or more sensors in the feature extraction as treated in e.g. [6], [15]. Such an extension of our method to multimodal sensors (e.g., image and sound sensors) will also be addressed in the near future.

APPENDIX

DETAILED DEFINITIONS OF THE FUNCTIONS U_1 AND U_2

Enjoying the compatibility of GPs, the function $U_1(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ which represents the mutual information is defined as

$$U_1(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = -\frac{1}{2} \log \left(\frac{\det \tilde{\boldsymbol{\Sigma}}}{\det \mathbf{S} \det \boldsymbol{\Sigma}} \right),$$

where $\tilde{\boldsymbol{\Sigma}}$ and S_{ab} which is the (a, b)-th element of $\mathbf{S}$ for $a, b = 1, 2, \ldots, d_{\mathbf{y}}$ are defined as follows:

$$
\begin{aligned}
\tilde{\boldsymbol{\Sigma}} &= \left[\begin{array}{cc} \mathbf{S} & \mathbf{C}^{\mathrm{T}} \\ \mathbf{C} & \boldsymbol{\Sigma} \end{array} \right], \\
S_{ab} &= \boldsymbol{\beta}_a^{\mathrm{T}} \mathbf{Q}_{ab} \boldsymbol{\beta}_b - m_a m_b \\
&\qquad + \delta_{ab} \left(\alpha_a^2 - \mathrm{Tr}((\mathbf{K}_a + \sigma_a^2 \mathbf{I})^{-1} \mathbf{Q}_{aa}) \right).
\end{aligned}
$$

Here, we define $\mathbf{C} = \boldsymbol{\Lambda} - \boldsymbol{\mu}\mathbf{m}^{\mathrm{T}}$, $m_a = \boldsymbol{\beta}_a^{\mathrm{T}} \mathbf{q}_a$, $\mathbf{m} = [m_1, m_2, \ldots, m_{d_{\mathbf{y}}}]^{\mathrm{T}}$, $\boldsymbol{\beta}_a = \mathbf{K}_a^{-1} \mathbf{y}_a$, and $\mathbf{q}_a$ with its i-th element q_{ai} for $i = 1, 2, \ldots, N$,

$$
\begin{aligned}
q_{ai} &= \alpha_a^2 \det\Big(\boldsymbol{\Sigma} \big(\mathbf{H}_a^{\mathbf{s}}\big)^{-1} + \mathbf{I} \Big)^{-\frac{1}{2}} \\
&\quad \times \exp\Big(-\tfrac{1}{2} \big(\boldsymbol{\mu} - \mathbf{s}^{(i)}\big)^{\mathrm{T}} \big(\boldsymbol{\Sigma} + \mathbf{H}_a^{\mathbf{s}}\big)^{-1} \big(\boldsymbol{\mu} - \mathbf{s}^{(i)}\big) \Big) \\
&\quad \times \exp\Big(-\tfrac{1}{2} \big(\mathbf{x} - \boldsymbol{x}^{(i)}\big)^{\mathrm{T}} \big(\mathbf{H}_a^{\mathbf{x}}\big)^{-1} \big(\mathbf{x} - \mathbf{x}^{(i)}\big) \Big).
\end{aligned}
$$

The matrix $\mathbf{Q}_{ab}$ is defined with its (i, j)-th element $Q_{ab,ij}$ for $i, j = 1, 2, \ldots, N$,

$$
\begin{aligned}
Q_{ab,ij} &= \alpha_a^2 \alpha_b^2 \det\Big(\big(\big(\mathbf{H}_a^{\mathbf{s}}\big)^{-1} + \big(\mathbf{H}_b^{\mathbf{s}}\big)^{-1} \big) \boldsymbol{\Sigma} + \mathbf{I} \Big)^{-\frac{1}{2}} \\
&\times \exp\Big(-\tfrac{1}{2} \big(\mathbf{s}^{(i)} - \mathbf{s}^{(j)}\big)^{\mathrm{T}} \big(\mathbf{H}_a^{\mathbf{s}} + \mathbf{H}_b^{\mathbf{s}}\big)^{-1} \big(\mathbf{s}^{(i)} - \mathbf{s}^{(j)}\big) \Big) \\
&\times \exp\Big(-\tfrac{1}{2} \big(\mathbf{s}^{ij} - \boldsymbol{\mu}\big)^{\mathrm{T}} \mathbf{R}^{-1} \big(\mathbf{s}^{ij} - \boldsymbol{\mu}\big) \Big) \\
&\times \exp\Big(-\tfrac{1}{2} \big(\mathbf{x} - \mathbf{x}^{(i)}\big)^{\mathrm{T}} \big(\mathbf{H}_a^{\mathbf{x}}\big)^{-1} \big(\mathbf{x} - \mathbf{x}^{(i)}\big) \Big) \\
&\times \exp\Big(-\tfrac{1}{2} \big(\mathbf{x} - \mathbf{x}^{(j)}\big)^{\mathrm{T}} \big(\mathbf{H}_b^{\mathbf{x}}\big)^{-1} \big(\mathbf{x} - \mathbf{x}^{(j)}\big) \Big),
\end{aligned}
$$

where, $\mathbf{s}^{(i)}$ and $\mathbf{x}^{(i)}$ are the i-th column of $\mathbf{Z}$ defined as Eq. (4), and

$$
\begin{aligned}
\mathbf{s}^{ij} &= \mathbf{H}_b^{\mathbf{s}} \big(\mathbf{H}_a^{\mathbf{s}} + \mathbf{H}_b^{\mathbf{s}}\big)^{-1} \mathbf{s}^{(i)} + \mathbf{H}_a^{\mathbf{s}} \big(\mathbf{H}_a^{\mathbf{s}} + \mathbf{H}_b^{\mathbf{s}}\big)^{-1} \mathbf{s}^{(j)} \\
\mathbf{R} &= \big(\big(\mathbf{H}_a^{\mathbf{s}}\big)^{-1} + \big(\mathbf{H}_b^{\mathbf{s}}\big)^{-1} \big)^{-1} + \boldsymbol{\Sigma}.
\end{aligned}
$$

The vector $\boldsymbol{\lambda}_a$, which is the a-th column of $\boldsymbol{\Lambda}$ for $a = 1, 2, \ldots, d_{\mathbf{y}}$ is defined as follows:

$$
\begin{aligned}
\boldsymbol{\lambda}_a = \sum_{i=1}^{N} \beta_{ai} q_{ai} & \left(\big(\mathbf{H}_a^{\mathbf{s}}\big)^{-1} + \boldsymbol{\Sigma}^{-1} \right)^{-1} \\
& \qquad \times \left(\big(\mathbf{H}_a^{\mathbf{s}}\big)^{-1} \mathbf{s}^{(i)} + \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} \right),
\end{aligned}
$$

where β_{ai} is the i-th element of $\boldsymbol{\beta}_a$.

Next, the function $U_2(\boldsymbol{\mu}_{t-1}, \boldsymbol{\Sigma}_{t-1}, \mathbf{y}_t, \mathbf{x}_t)$ is defined as follows:

$$
\begin{aligned}
\boldsymbol{\mu}_t &= \boldsymbol{\mu}_{t-1} + \mathbf{C}_t \mathbf{S}_t^{-1} (\mathbf{y}_t - \mathbf{m}_t), \\
\boldsymbol{\Sigma}_t &= \boldsymbol{\Sigma}_{t-1} - \mathbf{C}_t \mathbf{S}_t^{-1} \mathbf{C}_t^{\mathrm{T}},
\end{aligned}
$$

where the vector $\mathbf{m}_t$ and the matrices $\mathbf{S}_t$ and $\mathbf{C}_t$ are constructed by substituting $\mathbf{x}_t$ for $\mathbf{x}$. See [1] for the detailed derivation.

References

[1] H. Saal, J.-A. Ting, and S. Vijayakumar, "Active sequential learning with tactile feedback," in *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 2010, pp. 677–684.

[2] K. Hsiao, L. P. Kaelbling, and T. Lozano-Pérez, "Task-driven tactile exploration," in *Robotics Science and Systems*, 2010.

[3] B. Browatzki, V. Tikhanoff, G. Metta, H. Bulthoff, and C. Wallraven, "Active object recognition on a humanoid robot," in *IEEE International Conference on Robotics and Automation*, 2012, pp. 2021–2028.

[4] N. Lepora, U. Martinez-Hernandez, and T. Prescott, "Active touch for robust perception under position uncertainty," in *IEEE International Conference on Robotics and Automation*, 2013, pp. 3020–3025.

[5] A. Schneider, J. Sturm, C. Stachniss, M. Reisert, H. Burkhardt, and W. Burgard, "Object identification with tactile sensors using bag-of-features," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 243–248.

[6] J. Sinapov, C. Schenck, K. Staley, V. Sukhoy, and A. Stoytchev, "Grounding semantic categories in behavioral interactions: Experiments with 100 objects," *Robotics and Autonomous Systems*, vol. 62, no. 5, pp. 632–645, 2014.

[7] J. A. Fishel and G. E. Loeb, "Bayesian exploration for intelligent identification of textures," *Frontiers in Neurorobotics*, vol. 6, no. 4, 2012.

[8] C. E. Rasmussen, *Gaussian processes for machine learning*. MIT Press, 2006.

[9] N. Lawrence, "Probabilistic non-linear principal component analysis with Gaussian process latent variable models," *Journal of Machine Learning Research*, vol. 6, pp. 1783–1816, 2005.

[10] A. M. Okamura and M. R. Cutkosky, "Feature detection for haptic exploration with robotic fingers," *The International Journal of Robotics Research*, vol. 20, no. 12, pp. 925–938, 2001.

[11] S. Chitta, M. Piccoli, and J. Sturm, "Tactile object class and internal state recognition for mobile manipulation," in *IEEE International Conference on Robotics and Automation*, 2010, pp. 2342–2348.

[12] G. Heidemann and M. Schopfer, "Dynamic tactile sensing for object identification," in *IEEE International Conference on Robotics and Automation*, vol. 1, April 2004, pp. 813–818.

[13] M. Johnsson and C. Balkenius, "Experiments with proprioception in a self-organizing system for haptic perception," in *Towards Autonomous Robotic Systems 2007*, 2007, pp. 239–245.

[14] N. Gorges, S. Navarro, D. Göger, and H. Worn, "Haptic object recognition using passive joints and haptic key features," in *IEEE International Conference on Robotics and Automation*, 2010, pp. 2349–2355.

[15] 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.

[16] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Multifactor Gaussian process models for style-content separation," in *Proceedings of the 24th Annual International Conference on Machine Learning*, 2007, pp. 975–982.

[17] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *ICRA Workshop on Open Source Software*, 2009.

[18] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Learning attractor landscapes for learning motor primitives," in *Advances in Neural Information Processing Systems 15*, 2003, pp. 1523–1530.

[19] R. Storn and K. Price, "Differential evolution — a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[20] A. d'Avella, P. Saltiel, and E. Bizzi, "Combinations of muscle synergies in the construction of a natural motor behavior," *Nature neuroscience*, vol. 6, no. 3, pp. 300–308, 2003.

[21] T. Matsubara, S.-H. Hyon, and J. Morimoto, "Learning parametric dynamic movement primitives from multiple demonstrations," *Neural Networks*, vol. 24, no. 5, pp. 493–500, 2011.