Dexterous Manoeuvre through Touch in a Cluttered Scene

Wenyu Liang[1], Qinyuan Ren[2], Xiaoqiao Chen[2], Junli Gao[3], Yan Wu[1]

Abstract— **Manipulation in a densely cluttered environment creates complex challenges in perception to close the control loop, many of which are due to the sophisticated physical interaction between the environment and the manipulator. Drawing from biological sensory-motor control, to handle the task in such a scenario, tactile sensing can be used to provide an additional dimension of the rich contact information from the interaction for decision making and action selection to manoeuvre towards a target. In this paper, a new tactile-based motion planning and control framework based on bioinspiration is proposed and developed for a robot manipulator to manoeuvre in a cluttered environment. An iterative two-stage machine learning approach is used in this framework: an autoencoder is used to extract important cues from tactile sensory readings while a reinforcement learning technique is used to generate optimal motion sequence to efficiently reach the given target. The framework is implemented on a KUKA LBR iiwa robot mounted with a SynTouch BioTac tactile sensor and tested with real-life experiments. The results show that the system is able to move the end-effector through the cluttered environment to reach the target effectively.**

I. Introduction

Robots are increasingly expected to perform physical interaction and manipulation tasks in a given environment, such as grasping, pick-and-place, human-robot interaction [1], [2], etc. To achieve these dexterous and dynamic interactions with the environment, touch is crucial to close the control loop with adequate feedback [3], especially when visual cues are insufficient, unusable or even unavailable for the robot to make a good-enough informed decision for motion planning. One such example is the search mission under rubble. In this vision-denied densely-cluttered environment filled with a mixed range of objects such as rocks, sand, parts of daily objects, debris and human bodies, manoeuvre through the rubble effectively can only rely on the sense of touch.

Traditionally, touch is sensed at a single point, such as the end-effector, through the use of a force/torque sensor or estimated by joint torques. However, for complex manipulation tasks, far richer touch information (e.g., multiple point touch) is required for environment property inference and contact state estimation on the fly [4]. Recent advances in tactile perception show that tactile sensors which mimic the cutaneous receptors beneath the human skin, can be used to understand the interacted environment, such as surface textures [5], [6], edges [7], object pose [8], collisions [9], contact forces [10], contact configuration, slippage and object geometry [3]. This makes tactile sensing a complementary sensory modality to vision in providing environmental information for a manipulation task.

One challenge for tactile sensing as highlighted by Zou et al. [11] is how to develop an effective method to extract insights from the rich and complex tactile data, especially in unstructured environments. Many approaches have been proposed to acquire application-driven results using different machine learning techniques such as Decision trees and Naïve Bayes trees [12], Bayesian exploration [13], Support Vector Machine (SVM) [14], deep learning and active perception [15], [16], etc. These outcomes show that useful information can be extracted from tactile data using machine learning approaches specific to the tasks.

Apart from recognizing the environmental patterns, on-the-fly decision making based on the perceived patterns to generate a motion sequence is needed to enable the robot to manipulate the environment and manoeuvre to the target. For example, the manipulator needs to avoid or minimize the resistance from the environment applied to the robot when it is moving towards the target. This is a challenging motion planning problem because of the unavoidable multi-contacts and interactions with dense number of obstacles in the environment [17]. To our knowledge, limited research has been carried out on tactile-based motion planning for robot in clutters. In [18] and [19], model predictive control is used to address the reaching problem in a lightly cluttered environment. Moreover, motion control approaches, including feedback linearization control and adaptive control, are proposed to control a redundant manipulator in environments with uncertainty using tactile feedback [20], [21]. However, most approaches still require contact models which are difficult to build in a real-life unstructured environment. Thus, given the tactile information, a data-driven approach to build a model for motion plan generation can be potentially advantageous [22], [23].

In nature, primates are able to perform excellent actions to move their hands/fingers or manipulate objects in a cluttered environment although such environment is complex and unstructured. Therefore, an instinctive approach of controlling the robot manipulation is to adopt the biological motor control strategy. In this paper, a bioinspired hierarchical end-to-end motion planning and control framework leveraging on the information learned from the raw tactile data is developed. The proposed framework mainly focuses on the robot manipulation in a densely cluttered environment with the objective of effectively manoeuvre to a given target.

*This research is supported by the Agency for Science, Technology and Research (A*STAR) under its AME Programmatic Funding Scheme (Project #A18A2b0046).

[1] W. Liang and Y. Wu are with the Institute for Infocomm Research (I^{2}R), A*STAR, Singapore. liangw@i2r.a-star.edu.sg and wuy@i2r.a-star.edu.sg

[2] Q. Ren and X. Chen are with the College of Control Science and Engineering, Zhejiang University, Hangzhou, China. renqinyuan@zju.edu.cn and xqchen@zju.edu.cn

[3] J. Gao is with the School of Automation, Guangdong University of Technology, Guangzhou, China. jonygao621@gdut.edu.cn

The main contributions of this paper are: (i) propose and develop a bioinspired motion planning and control framework based on tactile feedback for robot manipulation in a densely cluttered environment, which offer a better solution than the conventional position control and impedance control methods; (ii) design an iterative two-stage data-driven approach for tactile state representation and manipulator motion planning. The rest of this paper is organized as follows. In Section II, the robotic system and the task are described. Then, the motion planning and control framework is presented in detail in Section III. Section IV presents the experimental results and discussions before conclusions are drawn in Section V.

II. Robotic System and Manipulation Task

The robot and experiment setup are shown in Fig. 1.

A. Robot Manipulator with Tactile Sensor

The robot system consists of a 7-degrees-of-freedom (7-DoF) KUKA LBR iiwa 14 R820 manipulator and a SynTouch BioTac tactile sensor secured by a 3D-printed end-effector. The iiwa robot offers high flexible motion and high-performance servo control with reliability of ± 0.15 mm. The BioTac sensor has 19 impedance sensing electrodes, 1 hydro-acoustic pressure sensor, and 1 thermistor [24], providing fine multi-modal sensing with a force resolution of 0.01 N and a pressure resolution of 0.0365 kPa.

B. Manipulation Task in a Densely Cluttered Environment

The robot system is required to perform a real-world manipulation task of manoeuvring to a target pose in a densely cluttered environment (e.g. a box of soft plastic balls). Fig. 2 illustrates different scenarios of this manoeuvre task. In an environment without unknown obstacles as depicted in Fig. 2(a), motion planning and control is straight forward. However, in a cluttered environment shown in Fig. 2(b), the robot will need to overcome obstacles that it comes into contact with to reach the target. Without an effective motion planning framework that considers the available tactile information, the robot may either at one extreme be unable to get the target at all, or at the other, damage itself and/or the environment by colliding into the obstacles to get to the target. This work focuses on finding a possible and

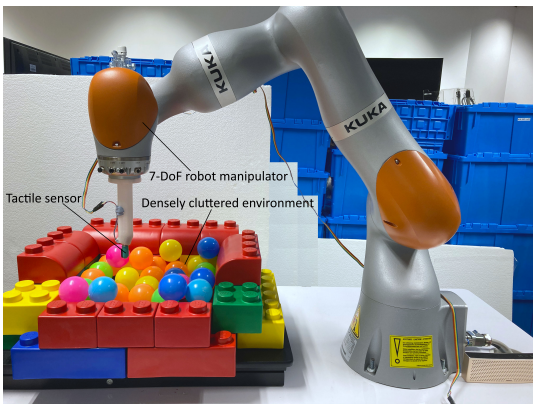

Fig. 1. System setup

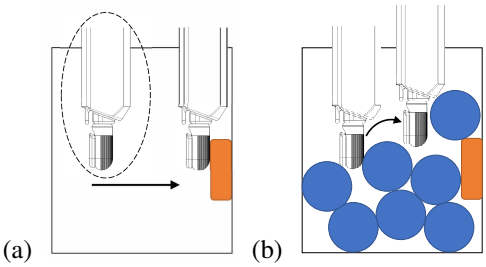

Fig. 2. Illustrations of a manipulation task in different environments (orange block denotes the target position; blue circles denote unknown obstacles): (a) structured environment (without obstacles); (b) cluttered environment (with motion planning to reduce resistance)

optimal motion/path based on the tactile information that can minimize the resistance in the cluttered environment while tracking the target as shown in Fig. 2(b).

III. Tactile-based Motion Planning and Control

In the biological sensory-motor system, the cerebral cortex provides concise representations of sensory state, context, and action while the basal ganglia are involved in action selection by evaluation of candidate actions [25], [26]. Inspired by the mechanism of the human nervous system and the work presented in [27], a tactile-based motion planning and control framework is proposed as depicted in Fig. 3.

The proposed framework mainly consists of three modules: tactile perception module, motion planning module and motion control module. Drawing from the investigation on biological system that the cerebral cortex is specialized for unsupervised learning while the basal ganglia are specialized for reward-based learning [25], autoencoder (AE) and reinforcement learning (RL) are employed as the learning mechanisms for the tactile perception module and the motion planning module, respectively. Therefore, the proposed framework has an outer feedback loop from the tactile sensing to the robot manipulation and two inner learning loops: one is within the tactile perception module and the other is between the agent and environment.

In a nutshell, the tactile sensing module receives the readings of the 19 sensing electrodes and makes an inference on the touch *state*. This *state* information is passed to the motion planning module which computes a corresponding *action*: the optimal motion commands at the current *state*. Finally, the motion control module regulates all the robot joint motors to achieve a desired *action*. Note that as this work focuses on developing the tactile perception and motion planning modules, the built-in position control mode of the robot is used as the motion control module.

A. Tactile Information Representation using Autoencoder

The AE, an unsupervised learning mechanism, is employed to extract the important features in a tactile instance without the need for explicit labels when the contact arises. It also reduces the input dimensionality to the motion planning module, allowing faster learning convergence of the module.

In this paper, a multilayer perceptron-based autoencoder (MLP-AE) is designed for representation learning because

the MLP is simple and efficient. It consists of an encoder and a decoder, which can be defined by the following transitions.

$$\left.\begin{aligned}\mathscr{E}&:\mathbf{X}\to\mathbf{Z}\\ \mathscr{D}&:\mathbf{Z}\to\mathbf{X}'\end{aligned}\right\}=\underset{\mathscr{E},\mathscr{D}}{\operatorname{argmin}}||\mathbf{X}-(\mathscr{E}\circ\mathscr{D})\mathbf{X}||^2, \tag{1}$$

where $\mathscr{E}$ and $\mathscr{D}$ represent the encoder and decoder, respectively, $\mathbf{X}$ is the input data, $\mathbf{Z}$ is the encoded data, and $\mathbf{X}'$ is the reconstructed data.

In the MLP-AE, the output layer has the same number of neurons as the input layer with the purpose of making the reconstructed data to be close to the input data. The designed MLP-AE is represented by the following equations.

$$\mathscr{E}:\begin{cases}\mathbf{h}=\psi(w^{(2)}\mathbf{x}+\mathbf{b}^{(1)})\\ \mathbf{z}=\psi(w^{(3)}\mathbf{h}+\mathbf{b}^{(2)}),\end{cases}\quad \mathscr{D}:\begin{cases}\mathbf{h}'=\psi(w^{(4)}\mathbf{z}+\mathbf{b}^{(3)})\\ \mathbf{x}'=\psi(w^{(5)}\mathbf{h}'+\mathbf{b}^{(4)}),\end{cases}$$

where $w^{(2)}\in\mathbb{R}^{l\times k}$, $w^{(3)}\in\mathbb{R}^{p\times l}$, $w^{(4)}\in\mathbb{R}^{l\times p}$, $w^{(5)}\in\mathbb{R}^{k\times l}$ are the weight matrices, $\mathbf{x}\in\mathbb{R}^k$ is the input vector, $\mathbf{z}\in\mathbb{R}^p$ is the encoded vector, $\mathbf{x}'\in\mathbb{R}^k$ is the reconstruction vector, $\mathbf{b}^{(1)}\in\mathbb{R}^l$, $\mathbf{b}^{(2)}\in\mathbb{R}^p$, $\mathbf{b}^{(3)}\in\mathbb{R}^l$, $\mathbf{b}^{(4)}\in\mathbb{R}^k$ are the bias vectors, $\mathbf{h}\in\mathbb{R}^l$, $\mathbf{h}'\in\mathbb{R}^l$ represent the hidden layer variables, and $\psi(\cdot)$ is the activation function and the logistic function is chosen:

$$\psi(x)=\frac{1}{1+\exp(-x)}, \tag{2}$$

where $\exp(\cdot)$ is the exponential function.

The AE is trained to minimize the reconstruction errors between the input and reconstructed data following a loss function based on the mean squared error (MSE) given by

$$L(\mathbf{X},\mathbf{X}')=\frac{1}{N}\sum_{n=1}^{N}||\mathbf{x}_n-\mathbf{x}'_n||^2, \tag{3}$$

where n denotes the n-th set of data, N is the length of data.

B. Reinforcement Learning-based Motion Planning

For the motion planning module, the main objective is to find the optimal motion sequence that can minimize the pressure and resistance applied on the end-effector while tracking the target in a short path. However, it is difficult to plan the motion via a model-based approach in an unstructured environment. To handle this problem, RL is employed because it offers a powerful model-free paradigm to solve for the optimal solution [28]. A neural network-based off-policy temporal difference (TD) reinforcement learning algorithm is used due to good data efficiency [29].

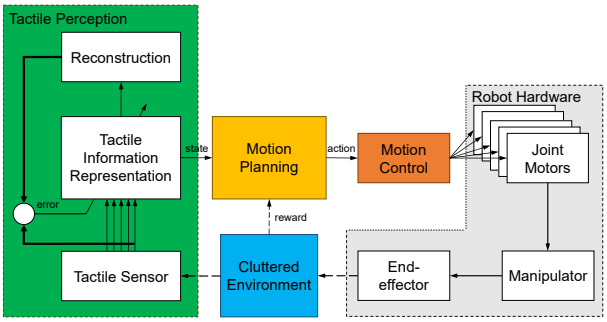

Fig. 3. An overview of the tactile-based motion planning and control system

1) Reinforcement Learning: The goal of learning is to find an optimal policy π^*, which tells the agent (i.e., the manipulator) to take an action a_k corresponding to different circumstances. By taking the action a_k, the agent transitions from state s_k to s_{k+1} and receives a reward r_{k+1} on (s_k,a_k), where the index represents discrete time k with $k\geq 0$. The optimal policy π^* is the policy that maximizes the expectation of the cumulative discounted reward R_k:

$$R_k=\sum_{t=0}^{T-k-1}\gamma^t r_{t+k+1}, \tag{4}$$

where γ is the discount factor and T is the final time step. In other words, the optimal policy can be described by $\pi^*=\arg\max_\pi \mathbb{E}[R_k|\pi]$, and π denotes the policy.

The objective of the manipulation task in this paper is to reach the target with minimum resistance. Hence, the reward function is defined by

$$r=-(\rho_1||p_{dc}||^2+\rho_2||d||^2), \tag{5}$$

where p_{dc} is the pressure output of the BioTac sensor which represents the resistance applied on the end-effector, d is the distance to the target, and ρ_1, ρ_2 are the weightings to determine the importance between the pressure and the distance. It is worth noting that higher reward can be earned if smaller p_{dc} and d are achieved.

In the designed RL, a Q-function measures the quality $Q^\pi(s_k,a_k)$ of taking a specific action a_k at a particular state s_k under a given policy π, where

$$Q^\pi(s_k,a_k)=\mathbb{E}[R_k|s_k,a_k,\pi]. \tag{6}$$

From (6), the optimal policy is then defined as a policy that maximizes the values of (6) over all possible state-action pairs (s,a). Thus, the optimal function Q^* is the maximum expected cumulative reward achievable from a given state-action pair (s_k,a_k): $Q^*(s_k,a_k)=\max_\pi Q^\pi(s_k,a_k)$. Since $Q^\pi(s_k,a_k)$ satisfies the Bellman equation, and we have

$$Q^*(s_k,a_k)=\mathbb{E}[(r_{k+1}+\gamma\max_{a'}Q^*(s_{k+1},a'))|s_k,a_k], \tag{7}$$

where a' is the possible action taken at state s_{k+1}.

2) Radial Basis Function Network-based RL (RBF-RL): To achieve the optimal function $Q^*(s_k,a_k)$, an artificial neural network (ANN) is used to represent the function (6), which can be obtained iteratively via the learning mechanism. A Radial Basis Function (RBF) network is chosen for the aforementioned purpose because it is stable, easy to design and generalize, and has simple structure, strong input noise tolerance and fast online learning ability [30].

Let $Q(s_k,a_k,\theta)$ be an approximation to the function (6) with a parameter θ, then the objective of the RBF network is to directly approximate the optimal function $Q^*(s_k,a_k)$: $Q(s_k,a_k,\theta)\approx Q^*(s_k,a_k)$. Such objective can be converted to an optimization problem of minimizing a loss function:

$$J_i(\theta_i)=\frac{1}{2}\mathbb{E}[(y_i-Q(s_k,a_k,\theta_i))^2], \tag{8}$$

where $y_i=r_{k+1}+\gamma\max_{a'}Q(s_{k+1},a',\theta_{i-1})$, i denotes the i-th episode.

The structure of the proposed RBF network is shown in Fig. 4, where the Gaussian functions in (9) are adopted,

$$\varphi_j(\mathbf{v}) = \exp\left(-\frac{||\mathbf{v}-\mu_j||^2}{\sigma_j^2}\right) \quad \text{for} \quad j = 1,2,...,m, \quad (9)$$

where $\varphi_j(\mathbf{v}) \in [0,1]$ denotes the distance of the input $\mathbf{v}$ to the jth region with the mean μ_j and the variance σ_{j,j_j}.

The input vectors $\mathbf{x} = \{x_1, x_2, \cdots, x_n\}$ are the state-action pairs. In this paper, the latent variables of the tactile perception module, the pose of the BioTac sensor are set as the states that represent the current circumstance of the fingertip. The action is defined as a vector of point-to-point motion indexes with different directions.

The RBF network output is a linear combination of the RBF outputs and the neuron weights, which is expressed by

$$Q(s_k, a_k, \theta_i) = \sum_{j=1}^{m} w_j \varphi_j = \mathbf{w}^T \phi, \quad (10)$$

where the parameter θ of the approximate Q-function is the weight vector $\mathbf{w}$, and $\phi(\mathbf{v}) = \begin{bmatrix}\varphi_1 & \varphi_2 & \dots & \varphi_m\end{bmatrix}^T$.

Finally, the weights in the RBF networks can be updated iteratively by applying the gradient descent algorithm to the loss function (8). The update rule is derived as follows:

$$\mathbf{w}_{i+1} = \mathbf{w}_i - \lambda \nabla_{\theta_i} J_i(\theta_i), \quad (11)$$

where $\nabla_{\theta_i} J_i(\theta_i) = -\left(y_i - Q(s_k, a_k, \theta_i)\right) \nabla_{\theta_i} Q(s_k, a_k, \theta_i)$.

From (10), the gradient $\nabla_{\theta_i} Q(s_k, a_k, \theta_i)$ at each weight can be obtained and the update rule is then given by

$$\mathbf{w}_{i+1} = \mathbf{w}_i + \lambda \left(y_i - Q(s_k, a_k, \theta_i)\right) \phi. \quad (12)$$

It is worth noting that the combination of the Q-learning with the neural network can lead to instability or divergence because the successive samples are correlated. To solve this problem, the following modified y_i is used [31], [32].

$$y_i = r_{k+1} + \gamma \max_{a'} Q(s_{k+1}, a', \theta_i^-), \quad (13)$$

where θ_i^- is the target parameter used to compute the Q-function at the i-th episode. This θ_i^- is only periodically updated with the neural network parameters θ_i every C steps, which is slowly changed and held fixed between individual updates.

Besides that, to balance the trade-off between exploitation and exploration, the ε-greedy exploration is applied.

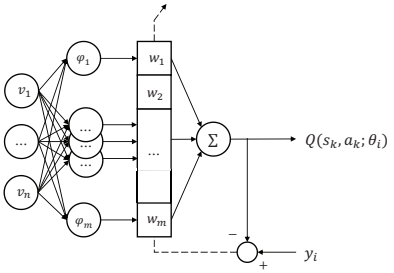

Fig. 4. Structure of the RBF network

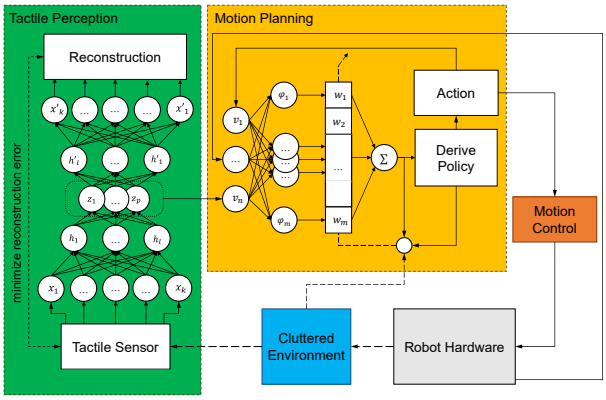

Fig. 5. Detailed overall control system

C. Integration of MLP-AE and RBF-RL

The full system, integrating the AE-based tactile perception module with the RL-based motion planning module, is shown in Fig. 5 in detail. Algorithm 1 gives the pseudo-code of the overall algorithm, where M is the final episode, C is a positive constant, the naive policy is a policy that moves the end-effector to the target in the shortest distance, and the two learning loops are trained in turn until satisfactory performance is achieved.

Algorithm 1 Online RBF-RL with MLP-AE

```
Pre-train MLP-AE network with a naive policy
Initialize RBF network weights θ and target weights θ⁻
while satisfied performance is not achieved do
    for episode i = 1 to M do
        Determine state s_k by encoding the sensor output
        for k = 1 to T do
            Select action a_k based on ε-greedy exploration
            Take action a_k, observe next state s_{k+1} and
            receive reward r_{k+1} = r
            y_i = { r,                                   for terminal s_{k+1}
                  { r + γ max_{a'} Q(s_{k+1}, a'; θ⁻),   otherwise
            Update θ according to (12)
            if k mod C == 0 then
                Update the target network: θ⁻ ← θ
            end if
            if target is reached then
                Break
            end if
        end for
    end for
    Re-train MLP-AE with the trained policy
end while
```

IV. Experimental Results and Discussions

Experiments are conducted in the setup shown in Fig. 1 to verify the effectiveness and performance of the proposed framework. In this setup, a box that is full of soft plastic balls with uniform size (diameter of about 50 mm) is used to create the densely cluttered environment. To implement the control system, a computer is connected to both the BioTac sensor driver and the iiwa controller via Robot Operating System

(ROS). The computer receives the sensor readings, computes the corresponding action through the proposed algorithm and sends the action command to the iiwa controller at 100 Hz.

A. Tactile Perception

To pre-train the MLP-AE, numbers of data are collected from the sensor while the manipulator moves to different positions in the shortest distance inside the ball pit. In total, 15000 sets of the 19 electrode outputs are used in the training and further 3000 sets of outputs are used for testing.

The MLP-AE is trained using the MATLAB Deep Learning Toolbox by setting the sizes of the hidden layer and the output layer of the encoder to be 10 and 5, respectively, i.e., $l = 10$, and $p = 5$. The scaled conjugate gradient backpropagation algorithm is used for the MLP-AE network training. The training process is shown in Fig. 6, where the loss function converges during 1500 iterations of training. Moreover, the goodness of fit on average measured by (14) is 95.36% for the training data and validation data.

$$R^2 = 1 - SS_{res}/SS_{tot} \tag{14}$$

where $SS_{res} = ||\mathbf{x}_n - \mathbf{x}'_n||^2$, $SS_{tot} = ||\mathbf{x}_n - \bar{x}_n||^2$, and $\bar{x}_n$ is the mean of $\mathbf{x}_n$, for $n = 1, \cdots, N$.

Figures 7 and 8 show the testing results on the trained MLP-AE. It can be observed that the trained MLP-AE can predict the testing data effectively, where the goodness of fit on average is 90.58%. It implies that the designed AE has good prediction performance and the encoder output well represents the tactile sensing information, i.e., up to 90% tactile sensing information can be represented by the 5 encoded latent variables instead of 19 electrode outputs, which greatly reduce the data dimension while the useful information is kept.

Furthermore, Fig. 9 visualizes the t-distributed stochastic neighbor embedding (t-SNE) [33] on the encoder output of some circumstances. It is clearly observed from the figure that the latent features of different circumstances (such as left contact, right contact, front contact and two contacts) can be separated in different clusters. Hence, it can be also concluded that the latent space of the MLP-AE is capable of extracting the relevant tactile features.

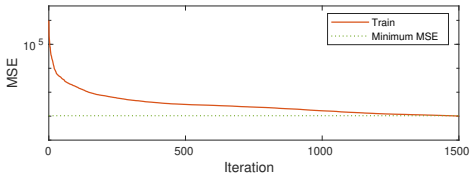

Fig. 6. MLP-AE training results

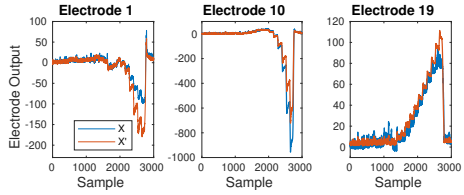

Fig. 7. MLP-AE testing results on three electrodes (blue: $\mathbf{x}_n$, red: $\mathbf{x}'_n$)

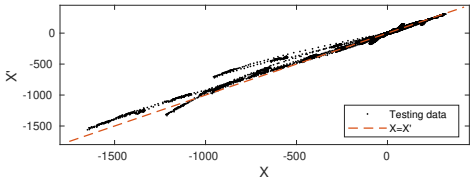

Fig. 8. MLP-AE regression performance

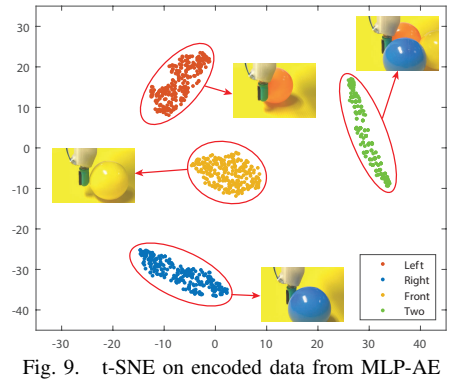

Fig. 9. t-SNE on encoded data from MLP-AE

B. Motion Planning

1) RBF-RL Training: To effectively sample from the vast state-action pairs in the experiment setup to train the RBF-RL network, we randomize the positions to insert the fingertip, the fingertip will then move according to the current policy until it reaches the target before repeating the process. The RBF-RL algorithm is trained based on a 50-episode cycle before the trained policy is used to re-train the MLP-AE state representation. During the MLP-AE re-training, 15000 sets of electrode outputs are used which consists of 40% of randomly selected old data and 60% of new data collected using the trained policy. The process is repeated until convergence occurs. In total, about 7800 sets of state-action pairs have been collected and used in the training of 300 episodes. By implementing the proposed RBF-RL algorithm, the training process and results are illustrated in Fig. 10. Both the cumulative reward (with a moving average of 25 episodes) and the norm of network weight vector $||\mathbf{w}||$ converge after about 95 episodes.

To benchmark the performance of the proposed approach, three other frameworks are used for comparison: (i) RBF-RL without MLP-AE, (ii) Tabular Q-learning with MLP-AE (TQ-RL with MLP-AE), and (iii) RBF-RL using pressure output instead of electrode outputs (P only). All these frameworks are trained with the same configurations. Fig.

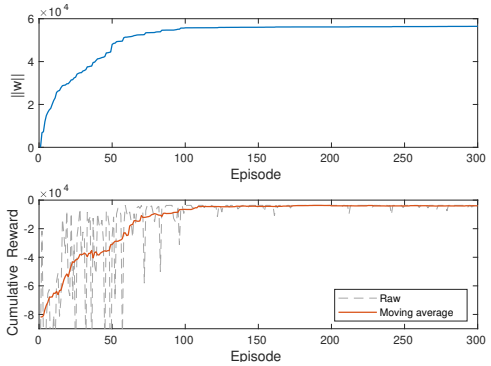

Fig. 10. Training process and results of the proposed framework

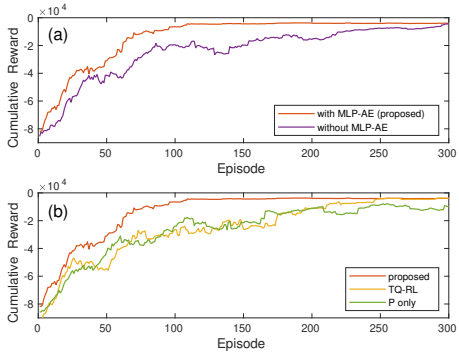

Fig. 11. Average cumulative rewards using different frameworks: (a) with MLP-AE vs. without MLP-AE; (b) comparisons of different frameworks

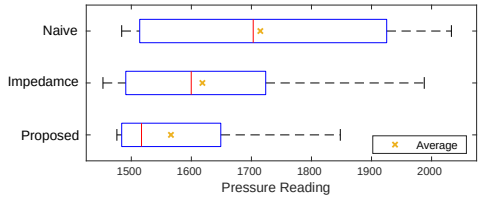

Fig. 12. Pressure comparison results

Fig. 13. Sequence of the manipulation task using the proposed framework

11 shows the comparisons on the average cumulative reward among different frameworks during the training process.

In Fig. 11(a), although the cumulative reward tends to be converged within the 300 episodes while the MLP-AE is not used, its convergence speed is slower than the integration of the MLP-AE and RBF-RL. The MLP-AE reduces the state space that goes into the RBF-RL, without which it may result in a much larger network requiring longer training time. The convergence of the integrated framework suggests that the MLP-AE succinctly and effectively compresses the states.

In Fig. 11(b), it can be found that the TQ-RL can also help the system to find the optimal policy in about 250 episodes. However, the convergence speed of using this traditional tabular method is much slower than the proposed framework. This can be mainly due to the curse of dimensionality [34]. Also, it is obvious that the framework using pressure output cannot achieve the maximum cumulative reward within the specified maximum episode. The comparison results between the framework only using pressure output and the one using tactile electrode outputs reveal that the tactile sensing information can help the system to learn the optimal policy in a shorter time. This is reasonable because the tactile electrodes can capture richer touch information comprehensively which is helpful in selecting the correct actions while the pressure output is just a kind of single point force measurement.

In summary, the proposed framework is effective and efficient, the strength of which lies in its ability to compactly represent both the high-dimensional tactile observations and the Q-function using neural networks.

2) Validation and Comparison: To further validate the effectiveness of the proposed framework, the learned framework is benchmarked in the real-time experiments. In the practical manipulation task, the end-effector is required to move forward to reach the target position surrounded by the plastic balls. Note that the task is considered as failed if the pressure output of the tactile sensor exceeds a common threshold or the end-effector is unable to reach the target position within an acceptable range. For the purposed framework, the trained RBF network are used and refined, and the robot action is selected based on the following criteria: $a_k = \arg\max_a Q(s_k, a; \theta^-)$. Moreover, two other motion planning and control strategies are used for comparison purpose: (i) naive policy: a straightly forward movement that moves the end-effector to the target position, and (ii) impedance control: the built-in impedance control mode of the robot is activated.

Ten tests are conducted using the proposed framework and the two other strategies, respectively. Fig. 12 shows a boxplot of the pressure applying on the tactile sensor with different approaches during the target reaching task.

It is obvious that the naive policy performs the worst because it forces its way to the target in the shortest path. In only 3 out of 10 tests, the end-effector managed to reach the target position. The failures are all due to pressure exceeding threshold. On the other hand, the impedance control and the proposed framework can reach the target position with higher success rates which are 60% for the impedance control and 100% for the proposed framework, respectively. In terms of the pressure readings shown in the boxplot, the proposed framework shows the smallest median pressure value as well as pressure variance, and achieves the best performance.

Thus, it can be concluded that the proposed framework is effective for the manipulation in the densely cluttered environment. Fig. 13 shows the sequence of manoeuvring the end-effector to reach the target position (behind the yellow plastic ball) in the cluttered environment using the proposed framework. With the proposed framework, the robotic end-effector can be guided to reach the given target efficiently with less resistance.

V. Conclusions

In this paper, a bioinspired motion planning and control framework based on tactile feedback is proposed and developed for the target reaching task by a robot manipulator in a cluttered environment. A two-stage machine learning approach for tactile feature extraction using an MLP-AE and manipulator motion planning leveraging on RBF-RL is designed. Experiments to verify and validate the proposed framework have been conducted with the scenario of a 7-DoF robot manipulator moving its end-effector to reach the given target through a ball pit. The results show that the proposed framework can effectively achieve the manipulation task with better performance than the conventional methods. However, more complicated cluttered scene is not tested in this paper. Some possible extension works that can be investigated in the future include enhancing the system robustness for more complex environments, and improving the perception-action loop in the control system.

REFERENCES

[1] A. Billard and D. Kragic, "Trends and challenges in robot manipulation," *Science*, vol. 364, no. 6446, p. eaat8414, 2019.

[2] P. Tsarouchi, S. Makris, and G. Chryssolouris, "Human–robot interaction review and challenges on task planning and programming," *International Journal of Computer Integrated Manufacturing*, vol. 29, no. 8, pp. 916–931, 2016.

[3] M. Prats, A. P. del Pobil, and P. J. Sanz, *Physical Interaction: The Contribution of Tactile Sensors*. Springer Berlin Heidelberg, 2013, pp. 111–128.

[4] Q. Li, O. Kroemer, Z. Su, F. F. Veiga, M. Kaboli, and H. J. Ritter, "A review of tactile information: Perception and action through touch," *IEEE Transactions on Robotics*, vol. 36, no. 6, pp. 1619–1634, 2020.

[5] T. Taunyazov, H. F. Koh, Y. Wu, C. Cai, and H. Soh, "Towards effective tactile identification of textures using a hybrid touch approach," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 4269–4275.

[6] R. Gao, T. Taunyazov, Z. Lin, and Y. Wu, "Supervised autoencoder joint learning on heterogeneous tactile sensory data: Improving material classification performance," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 10 907–10 913.

[7] N. F. Lepora, A. Church, C. De Kerckhove, R. Hadsell, and J. Lloyd, "From pixels to percepts: Highly robust edge perception and contour following using deep learning and an optical biomimetic tactile sensor," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2101–2107, 2019.

[8] Y. She, S. Dong, S. Wang, N. Sunil, A. Rodriguez, and E. Adelson, "Cable manipulation with a tactile-reactive gripper," in *The Robotics: Science and Systems 2020 (RSS)*. RSS, 2020, p. p029.

[9] D. De Gregorio, R. Zanella, G. Palli, S. Pirozzi, and C. Melchiorri, "Integration of robotic vision and tactile sensing for wire-terminal insertion tasks," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 2, pp. 585–598, 2018.

[10] Y. Wang, X. Wu, D. Mei, L. Zhu, and J. Chen, "Flexible tactile sensor array for distributed tactile sensing and slip detection in robotic hand grasping," *Sensors and Actuators A: Physical*, vol. 297, p. 111512, 2019.

[11] L. Zou, C. Ge, Z. J. Wang, E. Cretu, and X. Li, "Novel tactile sensor technology and smart tactile sensing systems: A review," *Sensors*, vol. 17, no. 11, p. 2653, 2017.

[12] N. Jamali and C. Sammut, "Majority voting: Material classification by tactile sensing using surface texture," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 508–521, 2011.

[13] J. A. Fishel and G. E. Loeb, "Bayesian exploration for intelligent identification of textures," *Frontiers in Neurorobotics*, vol. 6, p. 4, 2012.

[14] J.-P. Roberge, S. Rispal, T. Wong, and V. Duchaine, "Unsupervised feature learning for classifying dynamic tactile events using sparse coding," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 2675–2681.

[15] W. Yuan, C. Zhu, A. Owens, M. A. Srinivasan, and E. H. Adelson, "Shape-independent hardness estimation using deep learning and a gelsight tactile sensor," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 951–958.

[16] W. Yuan, Y. Mo, S. Wang, and E. H. Adelson, "Active clothing material perception using tactile sensing and deep learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1–8.

[17] N. Kitaev, I. Mordatch, S. Patil, and P. Abbeel, "Physics-based trajectory optimization for grasping in cluttered environments," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 3102–3109.

[18] A. Jain, M. D. Killpack, A. Edsinger, and C. C. Kemp, "Reaching in clutter with whole-arm tactile sensing," *The International Journal of Robotics Research*, vol. 32, no. 4, pp. 458–482, 2013.

[19] M. D. Killpack, A. Kapusta, and C. C. Kemp, "Model predictive control for fast reaching in clutter," *Autonomous Robots*, vol. 40, no. 3, pp. 537–560, 2016.

[20] C. Schuetz, J. Pfaff, F. Sygulla, D. Rixen, and H. Ulbrich, "Motion planning for redundant manipulators in uncertain environments based on tactile feedback," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 6387–6394.

[21] F. Sygulla, C. Schuetz, and D. Rixen, "Adaptive motion control in uncertain environments using tactile feedback," in *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, 2016, pp. 1277–1284.

[22] H. Van Hoof, N. Chen, M. Karl, P. van der Smagt, and J. Peters, "Stable reinforcement learning with autoencoders for tactile and visual data," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 3928–3934.

[23] C. Paxton, V. Raman, G. D. Hager, and M. Kobilarov, "Combining neural networks and tree search for task and motion planning in challenging environments," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 6059–6066.

[24] *BioTac® Product Manual*. SynTouch Inc., 2018.

[25] K. Doya, "What are the computations of the cerebellum, the basal ganglia and the cerebral cortex?" *Neural Networks*, vol. 12, no. 7-8, pp. 961–974, 1999.

[26] ——, "Complementary roles of basal ganglia and cerebellum in learning and motor control," *Current Opinion in Neurobiology*, vol. 10, no. 6, pp. 732–739, 2000.

[27] S. Lange and M. Riedmiller, "Deep auto-encoder neural networks in reinforcement learning," in *2010 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2010, pp. 1–8.

[28] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 6, pp. 2042–2062, 2018.

[29] L. Li, J. Li, L. Qin, J. Cao, M. S. Kankanhalli, and J. Zhu, "Deep reinforcement learning in soft viscoelastic actuator of dielectric elastomer," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2094–2100, 2019.

[30] H. Yu, T. Xie, S. Paszczynski, and B. M. Wilamowski, "Advantages of radial basis function networks for dynamic system design," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 12, pp. 5438–5450, 2011.

[31] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[32] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *2016 International conference on Machine Learning (ICML)*, 2016, pp. 1928–1937.

[33] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[34] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.