2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Touch-Based Manipulation with Multi-Fingered Robot using Off-policy RL and Temporal Contrastive Learning

Naoki Morihira[1], Pranav Deo[1], Manoj Bhadu[1], Akinobu Hayashi[1],
Tadaaki Hasegawa[1], Satoshi Otsubo[1] and Takayuki Osa[2,3]

***Abstract*— Tactile information holds promise for enhancing the manipulation capabilities of multi-fingered robots. In tasks such as in-hand manipulation, where robots frequently switch between contact and non-contact states, it is important to address the partial observability of tactile sensors and to properly consider the history of observations and actions. Previous studies have shown that Recurrent Neural Network (RNN) can be used to learn latent representations for handling observation and action histories. However, this approach is usually combined with on-policy reinforcement learning (RL) and suffers from low sample efficiency. Integrating RNN with off-policy RL could enhance sample efficiency, but this often compromises stability and robustness, especially as the dimensions of observation and action increase. This paper presents a time-contrastive learning approach tailored for off-policy RL. Our method incorporates a temporal contrastive model and introduces a surrogate loss to extract task-related latent representations, enhancing the pursuit of the optimal policy. Simulations and real robot experiments demonstrate that our proposed method outperforms RNN-based approaches.**

I. Introduction

Complex and dexterous manipulations akin to human capabilities have long posed a challenge for multi-fingered robots. Among the myriad of challenges in robotic manipulation, one of the most intricate tasks is ensuring a transition of an object to a desired state while adeptly alternating between contact and non-contact states. In recent years, RL has been applied to robot manipulation tasks involving multi-fingered robots and complex contact states [1]. These tasks either require a simulation environment with precise object pose access or a well-structured real-world setting equipped with multiple cameras for accurate target object tracking [2], [3]. One important element that has the potential to revolutionize the manipulation capabilities of multi-fingered robots is the use of tactile information. In recent years, significant advances have been made in this area. Because tactile sensors provide contact feedback even under visual occlusion, they are expected to enable robots to perceive and adapt to object states, albeit partially. However, using tactile sensors to solve tasks requires addressing their inherent partial observability of tactile sensors and developing strategies that adequately account for this limitation.

In situations with partial observability and uncertainties, details often remain ambiguous, aligning with the framework of partially observable Markov decision processe (POMDP).

[1]Honda R&D Co., Ltd. Innovative Research Excellence, Japan, {firstname_lastname}@jp.honda
[2]The University of Tokyo, Japan, osa@mi.t.u-tokyo.ac.jp
[3]RIKEN Center for Advanced Intelligent Project, Japan

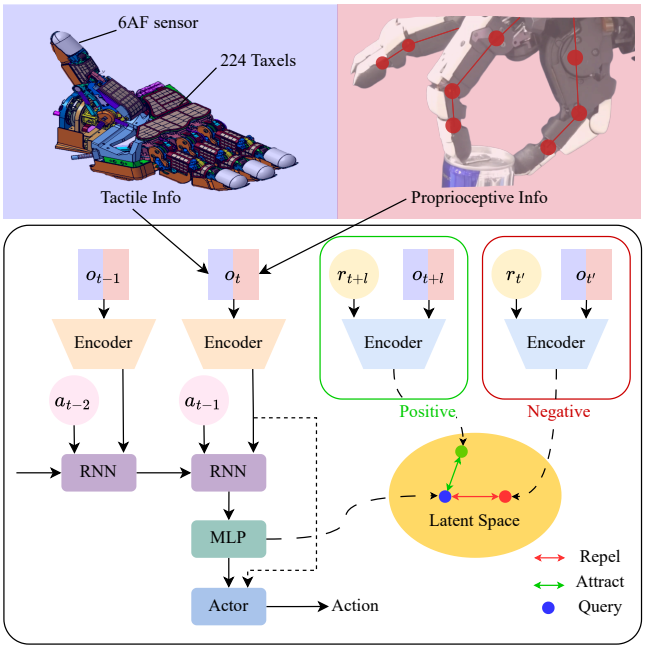


Fig. 1. We leverage tactile and proprioceptive data for touch-based manipulation with a multi-fingered robot. The latent representations of the history of observations and actions are learned to maximize mutual information through temporal contrastive learning.

Traditional methods, which rely on managing belief states, falter in high-dimensional spaces. Recent advances in RL have shown the potential of RNN to address this, but there are still gaps in off-policy RL, especially when it comes to tactile sensing in multi-fingered robot manipulation. The effectiveness of RNN diminishes with increasing dimensions, and they often require large sample sizes and extensive task-specific hyperparameter tuning [4]. Moreover, due to neural networks being nonlinear function approximations, there are convergence issues, leading to challenges in Q-value estimation. This underscores the limitations of RNN in off-policy RL settings, where sampling efficiency is paramount.

In this study, we present a framework that leverages temporal contrastive learning in off-policy RL, as depicted in Fig. 1. This approach drastically improves the sample efficiency on POMDP tasks. By incorporating a temporal contrastive model and introducing a surrogate loss, our method aids in extracting task-related representations, enhancing the pursuit of optimal policies. It addresses the challenges associated with relying on tactile feedback in multi-fingered robots.

Through rigorous evaluation, including extensive simula-

tions and experiments on real robot platforms, this work aims to demonstrate that the proposed method outperforms existing RNN-based approaches and contributes to the improvement of robot manipulation capabilities. The contributions of this paper are as follows:

1) We present a temporal contrastive learning approach that can be seamlessly integrated with existing off-policy RL algorithms to address the partial observability problem, suitable for both online and offline learning settings.
2) We empirically show in simulations and real-world experiments that on-policy and off-policy algorithms with general recurrent variants can handle low-dimensional tasks with partial observability, but fail in high-dimensional observation and action spaces.
3) Two demonstrations with a real-world robot that relies solely on tactile information and not vision show that the tactile-based manipulation approach can generalize robustly across different objects.

The remainder of this paper is organized as follows: Section II discusses related work with respect to representation learning and RNN-based RL algorithms. Section III explains our problem definition. Section IV introduces our approach, and Section V presents experiments and results. Finally, Section VI provides a conclusion and future work.

II. Related work

Addressing the intricacies of POMDP has been challenging due to the inherent partial observability [5], [6], where agents make decisions without full knowledge of the state of the environment. The integration of RNN has been a favored approach to address this challenge [4], [7]–[12]. However, directly incorporating RNN into an off-policy RL algorithm that relies on value-estimation bootstrapping for enhanced learning often leads to instability. A significant challenge arises when the learning process becomes more complex as the dimensions of observations and actions grow. This complexity arises from the simultaneous extraction of task-related representations and value function estimates.

Advances in representation learning, which extracts meaningful representations from high-dimensional unlabeled data, have been significant in fields such as language processing [13], [14] and computer vision [15]–[17]. These advances have been integrated into decision-making algorithms [18]–[20]. In addition, similar to our approach, several methods prioritize reducing the prediction error of future latent representations to capture task-specific representations [14], [21]–[24]. However, they often use identical or momentum-averaged encoders for both query and key observation encoding, potentially limiting their data comprehension capacity.

In the recent literature, numerous theoretical studies have explored the integration of representation learning with RL [25], [26]. These investigations have revealed certain design choices in representation learning that may not optimally benefit RL strategies. In our work, we have been careful to avoid making such inappropriate choices.

III. Problem Definition

A. POMDP

A POMDP [27] is defined as a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{P}, \mathcal{Z}, \mathcal{R}, \gamma \rangle$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $\mathcal{O}$ is the set of observations, $\mathcal{P} = \Pr(s_{t+1} \mid s_t, a_t)$ is the stochastic transition function, $\mathcal{Z} = \Pr(o_{t+1} \mid s_{t+1}, a_t)$ is the stochastic observation function, $\mathcal{R}(s_t, a_t)$ is the reward function, $\gamma \in [0, 1]$ is the discount factor, where t and $t+1$ represent successive time steps. The state, action, and observation at time step t are $s_t \in \mathcal{S}$, $a_t \in \mathcal{A}$, $o_t \in \mathcal{O}$. At each step, the agent receives a reward $r_{t+1} = \mathcal{R}(s_t, a_t)$. The goal is to compute the optimal policy π that maximizes the expected discounted total reward $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t)]$.

B. TD3

The twin-delayed deep deterministic policy gradient (TD3) algorithm is a model-free, online, off-policy RL method [28]. We consider the problem of optimizing a policy π_ξ parameterized by a vector ξ. In TD3, the Q-function is estimated with two function approximators Q_{ϕ_1} and Q_{ϕ_2}, with parameters ϕ_1 and ϕ_2. The parameters ϕ_i are learned by minimizing the Bellman error:

$$\mathcal{L}_{\text{critic}}(\phi_i) = E_{d\sim\mathcal{D}}\left[\left(Q_{\phi_i}(o_t, a_t) - (r_{t+1} + \gamma y)\right)^2\right], \quad (1)$$

where $d = \langle o_t, a_t, o_{t+1}, r_{t+1} \rangle$ is a tuple with observation o_t and next observation o_{t+1}, action a_t and reward r_{t+1}, $\mathcal{D}$ is the replay buffer and γ is the discount factor, and y is the target defined as

$$y = \min_{i=1,2} Q_{\phi_i'}(o_{t+1}, \pi_{\xi'}(o_{t+1}) + \varepsilon), \quad (2)$$

where $\varepsilon \sim \text{clip}(\mathcal{N}(0, \sigma), -c, c)$. Here, the added noise is clipped to keep the target close to the original action, and the parameters of $Q_{\phi_i'}$ and $\pi_{\xi'}$ are the exponential moving average (EMA) of the parameters of Q_{ϕ_i} and π_ξ, respectively. The use of EMA has been empirically shown to improve training stability in off-policy RL algorithms.

While the critic is given by Q_{ϕ_i}, the actor gets actions from the policy π_ξ and is trained by maximizing the expected return of actions given by

$$\mathcal{L}_{\text{actor}}(\xi) = E_{d\sim\mathcal{D}}\left[Q^\pi(o_t, \pi_\xi(o_t))\right]. \quad (3)$$

We utilize TD3 as a base RL algorithm combined with

temporal contrastive learning.

Algorithm 1 Off-policy RL with temporal contrastive learning (TCL)

Given:

- Total number of environment steps T

for each timestep $t = 1..T$ **do**
  Encode the observation $z_t = f_\psi(o_t)$
  Generate a query $q_t = h_\psi(z_{0:t}, a_{0:t-1})$
  Sample action $a_t \sim \pi_\xi(\cdot|z_t, q_t)$
  Execute the action $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$
  Observe the transition $o_{t+1} \sim \mathcal{Z}(\cdot|s_{t+1}, a_t)$
  Store the observed data $\mathcal{D} \leftarrow \mathcal{D} \cup (o_t, a_t, \mathcal{R}(s_t, a_t), o_{t+1})$
  Update the encoders using the Algorithm 2
  Update the critic by minimizing $\mathcal{L}_{\text{critic}}$
  Update the actor by maximizing $\mathcal{L}_{\text{actor}}$
end for

Algorithm 2 Encoder update by TCL

Given:

- Mini-batch size N
- Length of Backpropagation Through Time L_1
- Length of future prediction L_2

$(o_{t_i:t_i+L_1+L_2}, a_{t_i:t_i+L_1+L_2}, r_{t_i:t_i+L_1+L_2})_{i=1}^{N} \sim \mathcal{D}$
for each $i = 1..N$ **do**
  $z_{t_i:t_i+L_1+L_2} = f_\psi(o_{t_i:t_i+L_1+L_2})$
  $k_{t_i:t_i+L_1+L_2} = g_\psi(o_{t_i:t_i+L_1+L_2}, r_{t_i:t_i+L_1+L_2})$
  for each timestep $j = 1..L_1 + L_2$ **do**
    $q_{t_i+j} = h_\psi(z_{t_i:t_i+j}, a_{t_i:t_i+j-1})$
  end for
end for
Update the encoders by minimizing $\mathcal{L}_{\text{TCL}}$

IV. Method

In this section, we first provide an overview of the proposed framework and then describe its individual components.

A. Architecture Overview

We leverage Temporal Contrastive Learning (TCL) to extract task-related representations from time-series sequences of observations and actions. For tasks such as object manipulation, TCL facilitates the extraction of information related to object pose, particularly from past tactile sensor output and proprioceptive data. Our proposed method is detailed in Algorithm 1. Since we used TD3 as the base RL algorithm, we refer to the proposed method as TD3+TCL.

Unlike conventional approaches, which typically use identical or momentum-averaged encoders for query and key observation encoding, our strategy employs encoders with different parameterizations. This approach not only streamlines the integration of rewards into keys but also effectively filters out unnecessary constant elements from keys, such as target object poses. As shown in Fig. 2, we use a MultiLayer Perceptron (MLP) as an encoder to handle state inputs, while

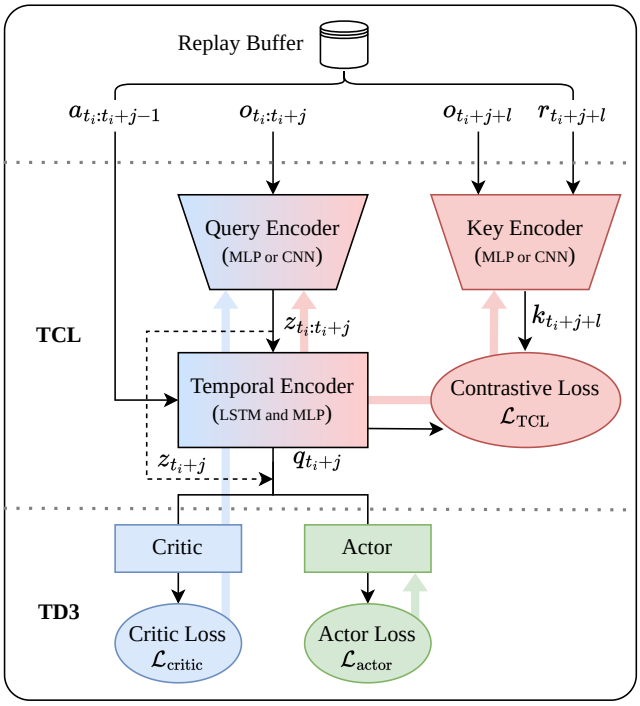


Fig. 2. Overview of the proposed framework.

image inputs are processed using a Convolutional Neural Network (CNN) for both query and key. The representations learned by TCL are given as inputs to both the actor and critic of the TD3 algorithm. The encoder and the policy are trained concurrently in our framework. In the offline setting, we employ TD3+BC [29] as our base RL algorithm and denote this configuration as TD3+BC+TCL.

B. Temporal Contrastive Learning

Contrastive learning aims to generate latent representations such that similar data points have similar features, while dissimilar data points are represented distinctly [15]. Conceptually, this can be seen as an optimization strategy to maximize the mutual information between data and their corresponding representations. In this study, we optimize the mutual information between a given query q, which aggregates past and current data sequences, and a key k, which has future data,

$$I(q;k) = \sum_{q,k} p(q,k) \log \frac{p(q|k)}{p(q)}. \tag{4}$$

First we use different encoders. For the key, it's defined as $k_t = g_\psi(o_t, r_t)$. For the query, it is a two-step process. Initially, the observation o_t is processed as $z_t = f_\psi(o_t)$. This output is then fed into a temporal encoder h_ψ that aggregates all $z_{0:t}$ and actions $a_{0:t-1}$, resulting in the contextual query $q_t = h_\psi(z_{0:t}, a_{0:t-1})$. By incorporating rewards into the key, we aim to embed the query, informed by observation and action histories, with essential task-related information. We used two layers of MLP as encoders for f_ψ and g_ψ, and a single layer of LSTM and MLP for h_ψ when the observation type is the state.

To maximize the mutual information between query and key, we apply the InfoNCE loss [14] during batch training. Given a set $K = \{k_1, ...k_N\}$ of N random samples with one positive sample k_{t+l} and $N-1$ negative samples, we compute the similarities between query and key with a log-bilinear model $q^t W k$. Thus, our InfoNCE loss is given by

$$\mathcal{L}_{\mathrm{C}}(\psi) = -\mathbb{E}_K \left[\log \frac{\exp(q_t^T W k_{t+l})}{\sum_{k_j \in K} \exp(q_t^T W k_j)} \right]. \quad (5)$$

To address the instability of the representations produced by the encoders, which are continuously updated via TCL during training, we introduced a regularization loss. This loss facilitates the consistency of the extracted representations by minimizing the mean squared error between the encoder outputs and those of a slowly changing target network. The number of dimensions for query and key are represented by d_q and d_k.

$$\begin{aligned}\mathcal{L}_{\mathrm{R}}(\psi) =& \frac{1}{d_q} \left\| h_\psi(f_\psi(o_{0:t}), a_{0:t-1}) - h'_{\psi'}(f'_{\psi'}(o_{0:t}), a_{0:t-1}) \right\|_2^2 \\ &+ \frac{1}{d_k} \left\| g_\psi(o_t, r_t) - g'_{\psi'}(o_t, r_t) \right\|_2^2 \end{aligned} \quad (6)$$

Here $f_{\psi'}$, $g_{\psi'}$ and $h_{\psi'}$ are the target networks and the parameters ψ' represent the exponential moving average (EMA) of parameters ψ. As a result, we train the encoders by minimizing the following objective function:

$$\mathcal{L}_{\mathrm{TCL}}(\psi) = \mathcal{L}_{\mathrm{C}}(\psi) + \alpha \mathcal{L}_{\mathrm{R}}(\psi). \quad (7)$$

Constant α balances the two terms in (7). In this work we fixed as $\alpha = 1$ for simplicity.

C. Training for Actor Critic

TD3 is designed for MDP environments, where the next state is determined solely by the current state and action, and the current state is directly observable. In POMDP settings, however, TD3 struggles to accurately estimate the Q-value from incomplete observations, leading to degraded performance and potential value estimation collapse.

To address this challenge, we incorporate representations generated by the temporal encoder for both actor and critic inputs. While the latent representations are trained to maximize mutual information with upcoming observations and rewards, they are implicitly bound to the current actual state. The policy receives two inputs: the observation encoded by the encoder f and the output of the temporal encoder h. The temporal encoder h takes the output of f and previous actions as inputs. Based on this, the policy generates actions as $a \sim \pi(a|f(o_t), h(f(o_{0:t}), a_{0:t-1}))$. Similarly, the critic receives the outputs of f and h in addition to the action and estimates the action value as $Q(f(o_t), h(f(o_{0:t}), a_{0:t-1}), a_t)$. By using the query from the temporal encoder, the policy and critic can incorporate the temporal information while performing the off-policy update.

As shown in Fig. 2, in our implementation, we decided to allow gradients from the critic loss to backpropagate through the query and temporal encoders to improve the extraction of representations directly relevant to expected returns.

V. Experiments

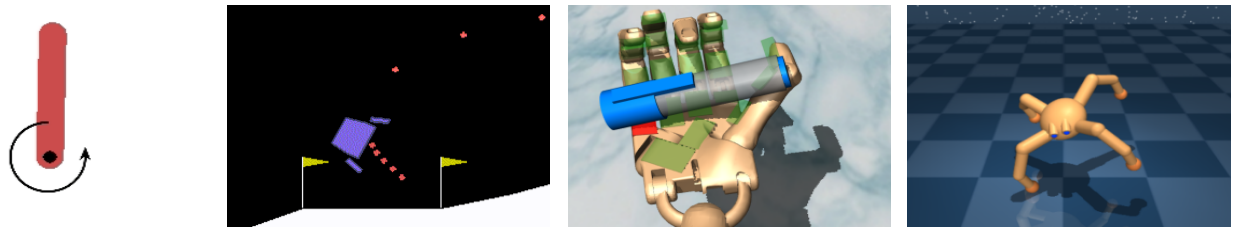
In this section, we present the results of benchmarks conducted on both simulators and a real robot. We used Proximal Policy Optimization (PPO) [30], Soft Actor-Critic (SAC) [31], and TD3 as our baseline methods. We adopted independent LSTM layers for both actor and critic components followed by two MLP layers [8]. This setup was relatively high-performing in our hyperparameter tuning, which included adjustments for various parameters such as LSTM unification and MLP layer count. We refer to the implementation of PPO, TD3, and SAC with their respective fully connected and LSTM versions, denoted as PPO(fc), PPO+LSTM, TD3(fc), TD3+LSTM, SAC(fc), and SAC+LSTM. For offline learning, we used TD3+BC and SAC+BC, which are modifications of TD3 and SAC designed for online learning. Their fully connected and LSTM versions are denoted as TD3(fc)+BC, TD3+BC+LSTM, SAC(fc)+BC, and SAC+BC+LSTM, respectively. Since the algorithms PPO(fc), TD3(fc), TD3(fc)+BC, SAC(fc), and SAC(fc)+BC operate within the framework of Markov Decision Process (MDP) environments, we used a stack of three observation frames as their inputs.

A. Test on Simulations

We chose diverse tasks (Fig. 3) including Pendulum (v1), Lunar Lander (v2) from OpenAI Gym [32], Pen (v0) from Adroit [33], and Quadruped Walk from DMC Vision [34] to cover various observation and action dimensions. The Pen task incorporated 21-dimensional tactile information highlighted in the green box in Fig. 3, and Quadruped Walk used image observations. We introduced partial observability by modifying the observations in each task: omitting velocity data in Pendulum and Lunar Lander, excluding object pose in Pen, and obscuring the observation image with 50% probability in Quadruped Walk [5]. In the Pen task, which aims at object reorientation, only joint angles and tactile data are used, reflecting our experimental setup with the real robot. Fig. 4 shows that MDP-based algorithms such as PPO(fc), TD3(fc), and SAC(fc) are effective in the Pendulum task, which has smaller dimensions. In the Lunar Lander task with slightly larger dimensions, our method, TD3+LSTM, and SAC-LSTM perform well. However, in high-dimensional tasks such as Pen and Quadruped Walk, only our method achieves successful learning. We used 1M steps of data for offline learning collected using TD3+TCL with online learning. As shown in Fig. 5, our method maintains relatively consistent performance, especially in larger dimensional environments. In Fig. 4 and Fig. 5, the shaded region represents the range of values within one standard deviation of the mean, calculated over 5 seeds.

B. Comparative Analysis of Latent Representations

In the Pen(touch) task, where only tactile and proprioceptive information are used as observations, we evaluated the latent representations of the TCL module in comparison to TD3+LSTM. Their alignment with the object pose is crucial for task performance. We collected 1M transition



Pendulum | Lunar Lander | Pen(touch) | Quadruped Walk(image)

Fig. 3. Dimensional overview of tasks. Pendulum (Observation: 1, Action: 1), Lunar Lander (Observation: 5, Action: 2), Pen (Observation: 51, Action: 24), and Quadruped Walk (Observation: Image 64x64x3, Action: 12).

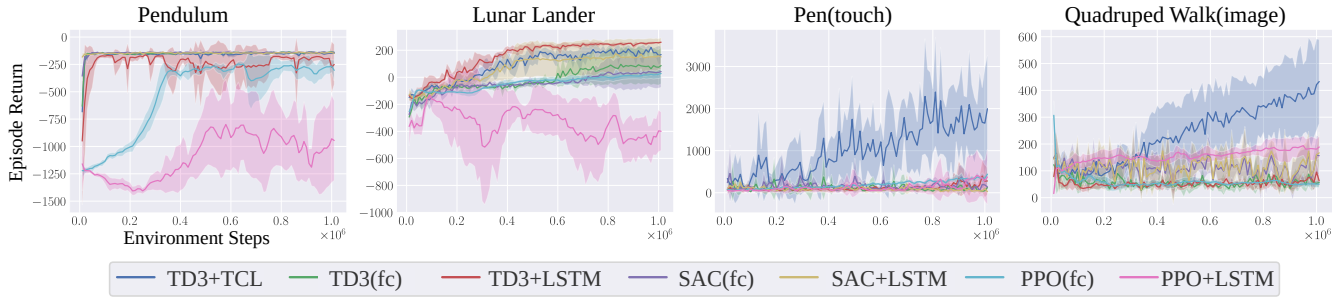


Fig. 4. Online learning results on simulator.

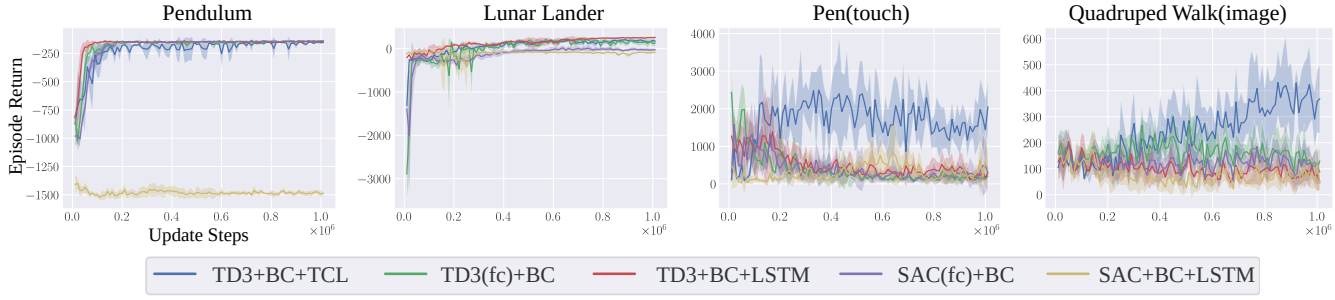


Fig. 5. Offline learning results on simulator.

data steps from both TD3+LSTM and TD3+TCL policies, each treated separately. For each policy, we trained a four-layer MLP regression model to estimate the object pose by minimizing the MSE between the predictions and the ground truth from the oracle (simulator), using the LSTM output representations as input. The encoder, LSTM unit size, and subsequent MLP were kept identical for each model. After training, we used these models to predict the object pose from the representations during rollouts of over 100k transition data steps that were not part of the training set.

The result of our study is shown in Fig. 6. It indicates a relatively strong correlation between the object pose information and the representations derived from TD3+TCL, emphasizing the effectiveness of our approach in capturing essential spatial details.

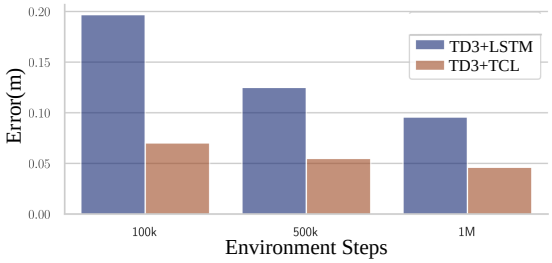


Fig. 6. Ablation study.

C. Test on Multi-fingered Robot

We used a wire-driven, multi-fingered robot developed by Honda R&D Co. Ltd. [35] in two different tasks. As shown in Fig. 7, this robot hand consists of four fingers. Each finger is equipped with a portion of the total 224 taxel tactile sensors [36], and a 6-axis force-torque sensor on the fingertip. This robot can apply up to 50N of fingertip force. A unique feature of this hand is its fingernail, which is designed to assist in tasks such as opening pull-tabs. For efficient benchmarking, we adopted the offline learning setting to compare various algorithms and seeds under data constraints. In these tasks, neither object pose nor vision information was provided. Instead, we relied on touch and proprioceptive data from the robot, which introduced complexity due to partial observability.

In the "Hook pull-tab" task, the goal is to hook onto the pull-tab of a can, as shown in Fig. 7. The position of the can varies randomly and uniformly within 1 cm along the x, y, and z axes. Task observations include force vectors and contact points for each finger, computed from 6-axis force-torque sensor values and fingertip shape, robot joint angles, and the action from the last step. The action space

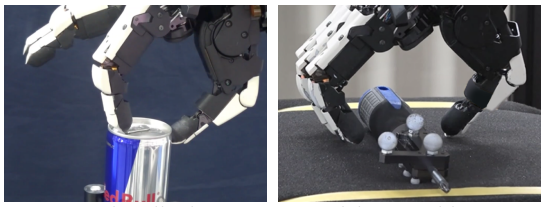


Fig. 7. Real robot setup.

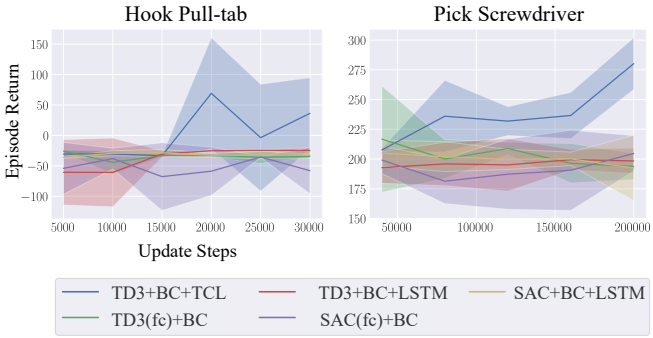


Fig. 8. Real robot experiment results.

is defined by the relative movement of the index finger in the task space. A sparse positive reward of 300 is given only if the finger successfully hooks onto the pull-tab. Success is determined by an upward force on the finger greater than 3.0N and whether or not the expected contact point is on the fingernail tip or not. Additionally, if the force norm exceeds 8.0 N, a negative penalty of -1.0 is imposed. Furthermore, we apply a penalty that is proportional to the action norm, with the action scaled within the range of -1.0 to 1.0. Thus, the episode return becomes positive only when the robot finger successfully hooks the pull-tab. Training data was collected from 5 hours of experiments with online TD3+TCL.

In the "Pick Screwdriver" task, the goal is to pick up a screwdriver from the state shown in Fig. 7. The policy receives observations that include force vectors and contact points for each finger (calculated using 6-axis force-torque sensor values and fingertip shape), tactile sensor values, the robot's joint angles, and the action from the last step. The action space is defined by desired joint angles. Rewards are calculated from the distance between the current and target pose of the screwdriver measured by motion capture system using markers attached to the screwdriver Fig. 7(right). The target pose of the screwdriver is specified as being in contact with the palmar surface of the hand when it is fully wrapped around the object being held. If the screwdriver is near the initial pose, the episode return is around 200, whereas if the screwdriver is successfully picked up at palm level, the episode return is close to 300. Training data was collected from 10 hours of experiments with online TD3+TCL.

As shown in Fig. 8, in scenarios where only proprioceptive and tactile data are available, our proposed method demonstrated superior performance in real robot experiments. In Fig. 8, the shaded region represents the range of values within one standard deviation of the mean, calculated over 5 seeds.

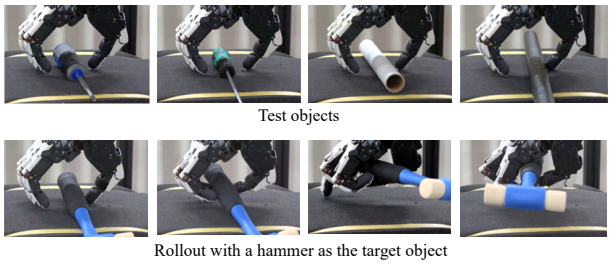


Fig. 9. Generalization across various objects.

TABLE I

SUCCESS RATE AGAINST UNSEEN OBJECTS

Objects	Success rate(%)
Driver(used for data collection)	40
Smaller driver	30
Core of wrap	40
Rack post	40
Hammer	50

D. Generalization Across Various Objects

In our task setting on the real robot, the observation space consists only of tactile and proprioceptive information. This strategy allows our model to effectively adapt to new objects that are similar to the training object. We validated this by applying the trained policy to cylindrical objects, as shown in Fig. 9, without any additional retraining. As shown in Table I, although the reward function was not ideally shaped, resulting in across-the-board lower performance, the success rates were nearly identical to that of the training object, demonstrating consistent performance across different unseen objects.

VI. CONCLUSION

In this paper, we introduced off-policy RL with TCL to enhance the manipulation capabilities of multi-fingered robots using touch information. Our method addresses the challenges of partial observability in tasks where the robot relies solely on tactile sensors. By integrating temporal contrastive learning with the off-policy RL algorithm, we were able to extract salient representations from sequences of observations and actions, enabling the robot to make informed decisions based on touch information alone.

Several key takeaways from our research include:

1) The potential of temporal contrastive learning to address the challenges of partial observability in robotic manipulation tasks.
2) The versatility of our method, which can be seamlessly integrated with existing off-policy RL algorithms.
3) The generalizability of our method, which allows it to perform effectively on objects of similar shape without retraining.

In future work, we aim to explore the integration of other sensory modalities with our method and to further improve its generalization capabilities across a broader range of objects and tasks. We believe that our research provides a solid foundation for the development of more advanced and versatile robotic manipulation systems.

REFERENCES

[1] M. T. Mason. Toward robotic manipulation. *Annu. Rev. Control. Robotics Auton. Syst.*, 1:1–28, 2018. I

[2] OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafał Józefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation. *CoRR*, 2018. I

[3] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations. In *Proceedings of Robotics: Science and Systems (RSS)*, 2018. I

[4] Tianwei Ni, Benjamin Eysenbach, and Ruslan Salakhutdinov. Recurrent model-free RL can be a strong baseline for many POMDPs. In *International Conference on Machine Learning*, pages 16691–16723. PMLR, 2022. I, II

[5] Matthew Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps, 2017. II, V-A

[6] Peter Karkus, David Hsu, and Wee Sun Lee. Qmdp-net: Deep learning for planning under partial observability, 2017. II

[7] Zhihan Yang and Hai Nguyen. Recurrent off-policy baselines for memory-based continuous control. *CoRR*, abs/2110.12628, 2021. II

[8] Zihan Ding. Popular-rl-algorithms. `https://github.com/quantumiracle/Popular-RL-Algorithms`, 2019. II, V

[9] Ilya Kostrikov. Pytorch implementations of reinforcement learning algorithms. `https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail`, 2018. II

[10] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. II

[11] Dongqi Han, Kenji Doya, and Jun Tani. Variational recurrent models for solving partially observable control tasks. *CoRR*, abs/1912.10703, 2019. II

[12] Maximilian Igl, Luisa M. Zintgraf, Tuan Anh Le, Frank Wood, and Shimon Whiteson. Deep variational reinforcement learning for pomdps. *CoRR*, abs/1806.02426, 2018. II

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. II

[14] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018. II, IV-B

[15] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. *CoRR*, abs/1911.05722, 2019. II, IV-B

[16] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. *CoRR*, abs/2006.07733, 2020. II

[17] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *CoRR*, abs/2006.09882, 2020. II

[18] Irmak Guzey, Ben Evans, Soumith Chintala, and Lerrel Pinto. Dexterity from touch: Self-supervised pre-training of tactile representations with robotic play, 2023. II

[19] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models, 2023. II

[20] Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning, 2020. II

[21] Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z. Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *CoRR*, abs/1611.05397, 2016. II

[22] Evan Shelhamer, Parsa Mahmoudieh, Max Argus, and Trevor Darrell. Loss is its own reward: Self-supervision for reinforcement learning. *CoRR*, abs/1612.07307, 2016. II

[23] Jinhua Zhu, Yingce Xia, Lijun Wu, Jiajun Deng, Wengang Zhou, Tao Qin, and Houqiang Li. Masked contrastive representation learning for reinforcement learning, 2020. II

[24] Andrea Banino, Adrià Puidomenech Badia, Jacob Walker, Tim Scholtes, Jovana Mitrovic, and Charles Blundell. Coberl: Contrastive bert for reinforcement learning, 2022. II

[25] Kate Rakelly, Abhishek Gupta, Carlos Florensa, and Sergey Levine. Which mutual-information representation learning objectives are sufficient for control?, 2021. II

[26] Yunhao Tang, Zhaohan Daniel Guo, Pierre Harvey Richemond, Bernardo Ávila Pires, Yash Chandak, Rémi Munos, Mark Rowland, Mohammad Gheshlaghi Azar, Charline Le Lan, Clare Lyle, András György, Shantanu Thakoor, Will Dabney, Bilal Piot, Daniele Calandriello, and Michal Valko. Understanding self-predictive learning for reinforcement learning, 2022. II

[27] Karl Johan Åström. Optimal control of markov processes with incomplete state information i. *Journal of mathematical analysis and applications*, 10:174–205, 1965. III-A

[28] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods, 2018. III-B

[29] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. IV-A

[30] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. V

[31] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. V

[32] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016. V-A

[33] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2020. V-A

[34] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018. V-A

[35] Tadaaki Hasegawa, Hironori Waita, Tomohiro Kawakami, Yoshinari Takemura, Tetsuya Ishikawa, Yuta Kimura, Chiaki Tanaka, Kenichiro Sugiyama, and Takahide Yoshiike. Powerful and dexterous multi-finger hand using dynamical pulley mechanism. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 707–713. IEEE, 2022. V-C

[36] Ryusuke Ishizaki, Shun Ogiwara, Fumiya Hamatsu, Tomoyuki Sakurai, Hirofumi Shin, and Takahide Yoshiike. Load-sensitive data acquisition for a tactile sensor system of multi-fingered robotic hands. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10767–10773. IEEE, 2022. V-C