2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

ConTact: Contrastive Tactile Alignment for Sim-to-Real Robotic Manipulation

Yanlin Lai*, Yinzhao Dong*†, Chun Yuan†, Cheng Zhou

***Abstract*— Deep reinforcement learning (DRL) has achieved remarkable success in robot control. However, DRL with tactile feedback still faces challenges in contact-rich tasks involving visual occlusion or high-speed dynamics. The challenges stem from two primary sources. First, the complexity and diversity of real-world tactile sensors make them difficult to simulate and transfer to reality. Second, existing high-fidelity simulators are often too computationally intensive for large-scale DRL, forcing a trade-off between accuracy and speed. To address this, we design a high-speed tactile simulation model for tactile arrays enabling efficient, large-scale DRL training on GPUs. We then propose the Contrastive Tactile (ConTact) framework, which leverages contrastive learning to align tactile features for sim-to-real transfer. ConTact employs a dedicated spatiotemporal encoder that explicitly models temporal changes to capture the dynamic features of contact events. We then validate it on two kinds of manipulation tasks, Single and Composite Object Tracking (SOT/COT), which rely solely on tactile information and proprioception. Moreover, policies trained with ConTact from simulation are directly deployed in the real world without finetuning, achieving zero-shot transfer.**

I. INTRODUCTION

Deep Reinforcement Learning (DRL) has emerged as a paradigm in robotic control, enabling robots to learn complex behaviors through environmental interaction and achieving remarkable success in various scenarios, such as agile locomotion [1], autonomous navigation [2], and dexterous manipulation [3]. However, these successes often rely on vision, which falters in contact-rich scenarios involving object occlusion, delicate handling, or high-speed sliding [4], [5]. Incorporating tactile feedback into DRL can improve performance in these contact-rich tasks. Thus, reliable tactile hardware, fast high-fidelity tactile simulation, and robust sim-to-real transfer are crucial for robot control.

*Equal Contribution. †Corresponding author: yinzhaodong@tencent.comyuanc@sz.tsinghua.edu.cn

Yanlin Lai is an intern at Tencent Robotics X, Shenzhen, Guangdong 518057, China, and is currently affiliated with Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, Guangdong 518057, China. laiyl24@mails.tsinghua.edu.cn

Yinzhao Dong is affiliated with Tencent Robotics X, Shenzhen, Guangdong 518057, China. yinzhaodong@tencent.com

Chun Yuan is affiliated with Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, Guangdong 518057, China. yuanc@sz.tsinghua.edu.cn

Cheng Zhou is affiliated with Tencent Robotics X, Shenzhen, Guangdong 518057, China. mikechzhou@tencent.com

This work was supported by the National Key R&D Program of China (2022YFB4701400/4701402), SSTIC Grant(KJZD20230923115106012, KJZD20230923114916032, GJHZ20240218113604008); This work was also sponsored by CIE-Tencent Robotics X Rhino-Bird Focused Research Program.

The supplementary video is available at https://tactilelabx.github.io/ConTact/

Reliable tactile hardware is the foundation for dexterous manipulation. Early sensors can measure global forces accurately [6], but they often fail to capture the pressure distribution and multi-point interactions. To address these challenges, researchers develop advanced tactile hardware, including tactile sensor arrays and artificial skins. However, the diverse sensing principles, such as resistive, capacitive, and vision-based [7], [8], lead to a lack of a universal simulation modeling paradigm.

Furthermore, a fast and reliable tactile simulation method is crucial for large-scale DRL training, providing massive tactile data while avoiding hardware drawbacks—such as high cost, fragility, and poor data stability [8]. Currently, mainstream simulated tactile sensors fall into two categories: Finite Element Method (FEM)-based (e.g., TacSL [9], TACTO [10], Taxim [11] and Taccel [12]) and penetration-based. While highly realistic, the intensive computational overhead of FEM makes it prohibitively slow for DRL training. To improve speed, MuJoCo MJX [13], IsaacGym [14], and IsaacLab [15] utilize the penetration depth of colliding geometries to estimate contact forces. However, large-scale deployment of tactile sensors is impractical in contact-rich tasks, as detecting all contact points imposes huge computational pressure on DRL training.

More importantly, bridging the sim-to-real gap remains a primary bottleneck for tactile-driven dexterous manipulation. Ding et al. [16] address this by simplifying tactile information to binary signals, which improves computational efficiency and bypasses the need for a precise tactile model. In contrast, some studies enhance tactile model robustness and high fidelity through domain randomization [17] and system identification [18]. However, for contact-rich tasks, the key features captured by real tactile sensors—such as contact force distribution, surface texture details, and object deformations—are critical for force-sensitive tasks, but are extremely difficult to accurately model in simulation.

To solve the above problems, we first design a computationally efficient tactile simulation model. Based on this, we then propose a pre-training Contrastive Tactile (ConTact) framework to align the tactile features, addressing the sim-to-real challenge in tactile signals. The key contributions of this work are listed as follows:

- We propose a fast ray-casting tactile simulation model in MuJoCo, boosting computational efficiency for large-scale DRL training on GPUs.
- We build a contrastive learning framework to bridge the sim-to-real gap by aligning tactile features between simulation and the real world.

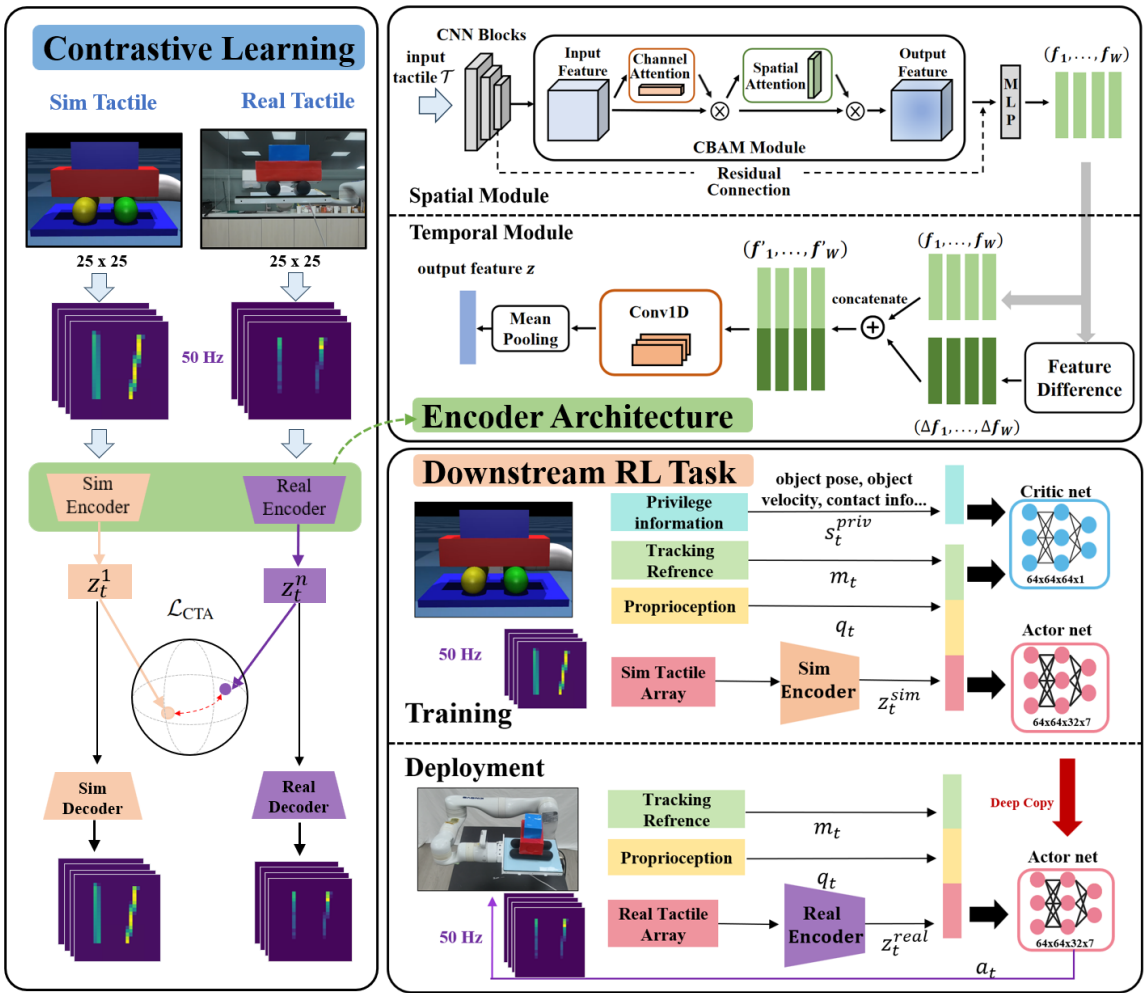


Fig. 1: The framework of our Contrastive Tactile Alignment method. (a) Our method first pre-trains tactile encoders using Contrastive Learning (left), where paired simulation and real-world tactile sequences are processed to align their latent representations via a contrastive loss (L_{CTA}). (b) The Encoder Architecture (top right) employs spatial and temporal modules to extract spatiotemporal features. (c) For the Downstream RL Task (bottom right), we train the policy in simulation using the Sim Encoder. This policy is then deployed zero-shot on the real robot, using the frozen, pre-trained Real Encoder to process real-time tactile feedback.

- We design a Spatial and Temporal module to extract the spatiotemporal feature, which is essential in dexterous and highly dynamic manipulation tasks.
- DRL controllers trained with our ConTact framework achieve zero-shot transfer to a physical robot, eliminating the need for real-world fine-tuning.

II. Method

Bridging the sim-to-real gap for tactile sensing is crucial for robust robotic manipulation. In this section, we first detail our novel high-speed tactile sensor simulation model, designed to capture essential contact mechanics for large-scale DRL training. Subsequently, we introduce the ConTact framework. As depicted in Fig. 1, ConTact leverages contrastive learning and a spatio-temporal encoder to align tactile features across simulated and real domains, extracting unified representations for downstream DRL tasks.

A. Tactile Sensor Array

1) Real world tactile sensor: Our experimental setup uses a real-world 25×25 tactile sensor array, as shown in Fig. 2(a). Each tactile element measures 8×8 mm, covering a total detection area of 20×20 cm. This grid structure enables the capture of pressure distributions at a fine resolution. As depicted in Fig. 2(c), the sensor surface is integrated with 1-mm-thick silicone and 3-mm-thick plastic foam layers, which provide mechanical compliance and allow it to conform to object surfaces. The sensor operates by detecting changes in resistance under applied pressure, outputting a real-time pressure map across a range of 500 Pa to 300 kPa.

2) Simulated tactile sensor: Our simulated tactile sensor is modeled in MuJoCo [13], specifically utilizing its JAX-based backend (MJX) for its high-speed, parallel processing capabilities on GPUs. Referring to the distribution of real-world tactile arrays, a series of detection sites is tiled D below the contact surface, as shown in Fig. 2 (b). Each site casts a normal ray with a maximum length of L. The

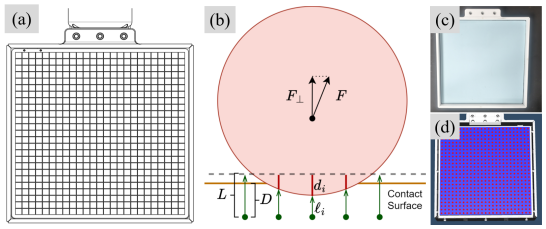


Fig. 2: (a) The 25×25 real-world tactile sensor array. (b) Our simulation model utilizes ray-casting; contact forces are distributed across detection sites based on their pseudo-penetration depth d_i. (c) real-world tactile tray, and (d) simulated tactile tray.

pseudo-penetration depth d_i of tactile site i is computed as: $d_i = L - \ell_i$, where ℓ_i is the ray's measured distance to the first collision. Notably, when the ray length matches the site depth ($L = D$), d_i becomes equivalent the normal object's penetration depth. By setting $L > D$, our model can capture pre-contact proximity perception, providing richer data for control.

While pseudo-penetration depth is geometrically informative, it is not a direct measure of force. To create a physically plausible tactile representation, our tactile model directly utilizes the contact force[1] computed by the MuJoCo physics engine. Specifically, we take the normal component of the contact force, $F_\perp$, and distribute it across the tactile sites. The force f_i assigned to each site is weighted by its pseudo-penetration depth d_i, ensuring that sites with greater penetration register proportionally higher force[2]:

$$f_i = w_i \cdot \|F_\perp\|, \quad w_i = \begin{cases} \dfrac{d_i}{\sum_j d_j} & \text{if } \sum_j d_j \neq 0, \\ 0 & \text{if } \sum_j d_j = 0. \end{cases} \tag{1}$$

Our goal is not to perfectly replicate the real sensor's electrical signals, which are influenced by noise and materials. Instead, we aim to provide a computationally efficient simulation that captures the essential physical dynamics of contact events. This model provides the robust simulation-side foundation required by our subsequent contrastive framework to effectively bridge the sim-to-real gap.

B. Contrastive Tactile Alignment Framework

This section elaborates on key components of ConTact framework: the generation of paired sim-to-real data, the contrastive learning approach for feature alignment, and the design of the spatio-temporal tactile encoder.

1) Pair Data Generation: To generate paired data for contrastive learning, we record object-tray interaction data in both simulation and the real world. For real-world data collection, the robot arm tracks a set of predefined trajectories using a Proportional-Derivative (PD) controller at a control rate of 50 Hz. We apply the same protocol in simulation, but on a much larger scale, to create a comprehensive dataset that broadly covers the real-world interactions. As detailed in Algorithm 1, we use a sliding window over the object's pose data P_t, obtained via a motion capture system, to compute the motion feature vector. The corresponding tactile data is denoted as T_t. The motion feature $\phi(\mathbf{P}_t)$ based on the poses of the objects over the window, $\mathbf{P}_t = [P_t, \ldots, P_{t+W-1}]$:

$$\begin{aligned} \phi(\mathbf{P}_t) &= \left(\mathbf{s}_t^{(1)}, \ldots, \mathbf{s}_{t+W-1}^{(1)}, \mathbf{s}_t^{(2)}, \mathbf{s}_{t+1}^{(2)}, \ldots, \mathbf{s}_{t+W-1}^{(K)}\right), \\ \mathbf{s}_t^{(k)} &= \left(\alpha \cdot \mathbf{p}_t^{(k)}, \ \beta \cdot \mathbf{v}_t^{(k)}, \ \beta \cdot \mathbf{u}_t^{(k)}\right), \end{aligned} \tag{2}$$

where $\mathbf{s}_t^{(k)}$ is the feature vector for object k at timestep t. It concatenates the object's position ($\mathbf{p}_t^{(k)}$) and its forward and up directional vectors ($\mathbf{v}_t^{(k)}, \mathbf{u}_t^{(k)}$). The scalars α and β serve as weight coefficients for position and orientation, respectively. With this feature representation, we then build a KD-tree on the simulation features to efficiently find the best-matching (i.e., the nearest neighbor) simulated tactile sequence for each real-world sequence, ensuring temporal and spatial consistency between pairs.

Algorithm 1 Sim-Real Paired Data Generation

Define: Sliding window size W, the number of tracked objects K, the size of dataset L_{sim} and L_{real};
Build dataset: simulated dataset $\{P_t^{sim}, T_t^{sim}\}_{t=0}^{L_{sim}}$, real dataset $\{P_t^{real}, T_t^{real}\}_{t=0}^{L_{real}}$;
for all dataset $\mathcal{D} \in \{\text{real}, \text{sim}\}$ **do**
  for $t = 0$ **to** $L_{\mathcal{D}} - W$ **do**
    Pose sequence: $\mathbf{P}_t^{\mathcal{D}} = [P_t^{\mathcal{D}}, \ldots, P_{t+W-1}^{\mathcal{D}}]$
    Extract motion feature: $g_t^{\mathcal{D}} \leftarrow \phi(\mathbf{P}_t^{\mathcal{D}})$
    Tactile sequence: $\mathcal{T}_t^{\mathcal{D}} = [T_t^{\mathcal{D}}, T_{t+1}^{\mathcal{D}}, \ldots, T_{t+W-1}^{\mathcal{D}}]$
  end for
end for
Build KD-tree using $\{g_j^{\text{sim}}\}$ for $j \in [0, L_{\text{sim}} - W]$
For each real window index $t \in [0, L_{\text{real}} - W]$:
  Query KD-tree with g_t^{real} to find best-matched j
  Form pair ($\mathcal{T}_t^{\text{real}}$, $\mathcal{T}_j^{\text{sim}}$)
Output paired dataset:
  Real tactile sequences: $[n, W, 25, 25]$
  Simulated tactile sequences: $[n, W, 25, 25]$

2) Contrastive Learning with Reconstruction: We employ a symmetric contrastive loss [19] to align the latent representations of paired data from different domains, preserving spatio-temporal information. For a batch of N paired tactile arrays, the loss is calculated bidirectionally. For instance, the loss for aligning the simulation data (domain A) to the real-world data (domain B) is:

$$\mathcal{L}_{A \to B} = -\frac{1}{N} \sum_{i=1}^{N} \log \frac{\exp(s(z_i^A, z_i^B)/\tau)}{\sum_{j=1}^{N} \exp(s(z_i^A, z_j^B)/\tau)}, \tag{3}$$

where $s(\cdot, \cdot)$ is the cosine similarity, z^A and z^B are latent representations, and τ is a temperature parameter. The to-

[1]The contact force is between two colliding objects, not object and tactile sensors. The rigid body dynamic simulation system can directly provide this force, without further computation burden.

[2]This model naturally extends to multi-point contact scenarios. The simulation engine first identifies all distinct contact pairs and their rays (identified by body IDs). We then apply the distribution process described in Eq. (1) to each contact pair independently and then aggregate the resulting forces at each correspond tactile site. This superposition principle allows us to robustly model complex interaction patterns.

tal Contrastive Tactile Alignment loss $\mathcal{L}_{\text{CTA}}$ averages both alignment directions:

$$\mathcal{L}_{\text{CTA}} = \frac{1}{2}\left(\mathcal{L}_{A\to B} + \mathcal{L}_{B\to A}\right), \tag{4}$$

where $\mathcal{L}_{B\to A}$ is from swapping the A and B in Eq. (3).

We also introduce a reconstruction loss for both domains. Since consecutive tactile frames are highly correlated, directly reconstructing them can fail to capture subtle dynamics. We therefore predict the initial frame I_1 and subsequent frame-to-frame differences $\Delta I_t = I_t - I_{t-1}$. The loss is a weighted Mean Squared Error:

$$\mathcal{L}_{\text{recon}} = w_0\|\hat{I}_1 - I_1\|_2^2 + \sum_{t=2}^{W} w_\Delta \|\Delta\hat{I}_t - \Delta I_t\|_2^2, \tag{5}$$

where $[\hat{I}_1, \Delta\hat{I}_t]_{t=2}^{W}$ is the decoded output, and weights $w_\Delta > w_0$ prioritize temporal dynamics.

Our final training objective combines both components with a balancing hyperparameter λ: $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{recon}} + \lambda\mathcal{L}_{\text{CTA}}$. This composite loss produces a feature space that is both robust to domain shifts and rich in contact dynamics.

3) Tactile Encoder-Decoder Architecture: We now detail the architecture of the spatio-temporal encoder used for feature extraction and the corresponding decoder used for reconstruction. We design the tactile encoder to process sequences of tactile images $(W, 25, 25)$ and generate a compact latent representation. The architecture is modular, consisting of a Spatial Module that processes individual frames and a Temporal Module that aggregates information across time.

a) Spatial Module: This module extracts a feature vector $f_t \in \mathbb{R}^D$ from each $(25, 25)$ tactile frame using Convolutional Neural Network (CNN). To enhance feature quality, it incorporates two key elements: a Convolutional Block Attention Module (CBAM) [20] to adaptively focus on salient channels and spatial locations, and a residual connection (Res) to facilitate training and preserve feature integrity. It processes each frame independently, producing the input sequence $(f_1, \ldots, f_W)$ for the temporal module.

b) Temporal Module: This module aggregates the feature sequence $(f_1, \ldots, f_W)$ to capture contact dynamics. We first compute a difference vector:

$$\Delta f_t = \begin{cases} \mathbf{0} & \text{if } t = 1, \\ f_t - f_{t-1} & \text{if } t > 1, \end{cases} \tag{6}$$

and concatenate it with the original feature to form an augmented vector $f'_t = [f_t, \Delta f_t]$. This Feature Difference (FD) module explicitly encodes temporal dynamics, rather than relying on the network to learn them implicitly.

The resulting sequence of augmented features is subsequently processed by a lightweight 1D convolutional network (Conv1D) to efficiently capture local temporal patterns. Finally, mean pooling is applied across the time dimension to produce a single, fixed-size latent vector z that represents the entire spatio-temporal tactile observation.

The decoder, active only during pre-training, complements the encoder by reconstructing the tactile sequence from the latent representation, utilizing transposed convolutional layers and interpolation to recover the original resolution.

III. Tactile-Aware Manipulation Controller

Motion tracking with DRL provides explicit sub-goals for policy learning, which reduces the exploration space and guides robots to learn more natural movements [21]. In the past decade, this technology has achieved significant performance in contact-rich tasks. For example, OmniH2O [22] teaches a robot to master tool-using skills, such as writing, sweeping, and watering plants. Cheng et al. [23] enable robots to learn diverse insertion and extraction tasks, including handling test tubes and recycling cans. Therefore, we develop two motion tracking manipulation tasks to evaluate the simulation tactile model and the ConTact framework. In the following, we will elaborate on the design of the tracking tasks and the details of DRL.

A. Task Specification

We introduce two kinds of manipulation tasks: Single Object Tracking (SOT) and Composite Object Tracking (COT). In both scenarios, a robotic arm equipped with a tactile tray end-effector must track a predefined reference motion. The core challenge is to maintain the object's stability throughout the trajectory, relying solely on the learned tactile features to infer the object's state without any visual input.

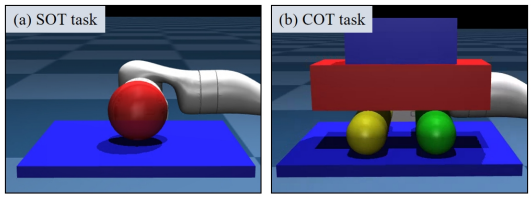


Fig. 3: Single Object Tracking and Composite Object Tracking task.

1) Single Object Tracking: The SOT task, shown in Fig. 3(a), requires the robot to keep a spherical object centered on the tactile tray, preventing it from falling off. This task serves as a baseline to evaluate the fundamental capability of our learned tactile representation to infer the object's position and physical properties.

2) Composite Object Tracking: The COT task in Fig. 3(b) presents a more advanced challenge: manipulating a composite toy car, which consists of a box-like body and two capsule wheels. Here, the controller not only tracks the car's overall position but also maintains its structural integrity. **The key challenge lies in detecting the asymmetric loading on the wheels that occurs when the box's center of mass shifts.** To prevent the collapse, the controller must leverage our learned features to infer the object's overall composite shape and react to pressure imbalances with corrective actions.

B. DRL Task Settings

To learn the policies for our object tracking tasks, we employ a DRL approach. We now define the key components: observation, action, and reward.

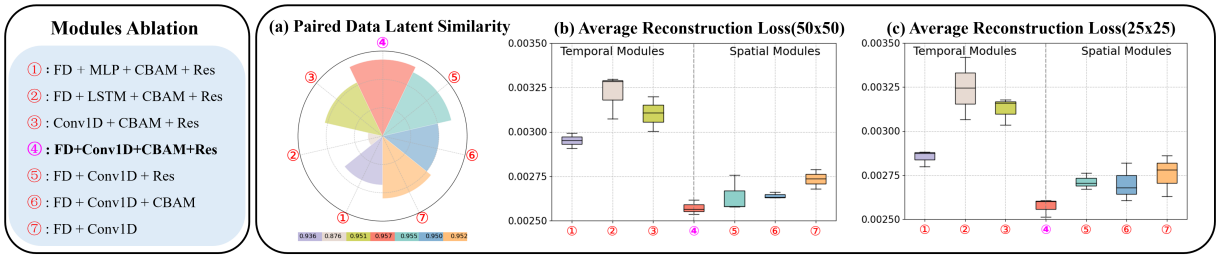


Fig. 4: Ablation study on the encoder architecture, evaluating latent similarity (a) and reconstruction loss (b-c).

a) Observation Space: The observation $\boldsymbol{o}_t$ consists of three components: the robot's proprioceptive state, a reference signal, and the learned tactile features. Specifically, it includes the current joint positions $\boldsymbol{q}_t \in \mathbb{R}^7$, the reference joint positions $\boldsymbol{m}_t \in \mathbb{R}^7$, a time phase variable $\psi_t \in [0, 1]$, and the latent tactile representation $\boldsymbol{z}_t$. The time phase ψ_t progresses from 0 to 1 over the trajectory's duration. The complete observation vector is formulated as:

$$\boldsymbol{o}_t = [\boldsymbol{q}_t, \boldsymbol{m}_t, \psi_t, \boldsymbol{z}_t]. \tag{7}$$

b) Action Space: The action $\boldsymbol{a}_t \in \mathbb{R}^7$ is a residual adjustment to the reference, yielding a target joint configuration $\boldsymbol{q}_t^* = \boldsymbol{m}_t + \boldsymbol{a}_t$. This target is then tracked by a low-level Proportional-Derivative controller.

c) Reward Function: As detailed in TABLE I, our reward function is structured to promote three key behaviors: tracking the reference motion, performing the specific balancing task, and ensuring smooth actions. Following the methodology of DeepMimic [21], most terms are formulated as exponential penalties to encourage precise control.

We employ an asymmetric actor-critic framework. The actor (policy) receives the standard observation $\boldsymbol{o}_t$ (Eq. 7) and can be zero-shot transferred to the real world. The critic, however, receives an augmented state that includes $\boldsymbol{o}_t$ and privileged information $\boldsymbol{s}_t^{priv}$, such as ground-truth object kinematics and contact physics. We use Proximal Policy Optimization (PPO) [24] to train both networks.

TABLE I: Reward Function Components and Weights.

Reward	Term	Equation	Weight
Tracking	Joint position tracking	$e^{-2\|\|\sum_j q^j - \hat{q}^j\|\|^2}$	0.5
	Arm position tracking	$e^{-5\|\|\sum_j p^j_{arm} - \hat{p}^j_{arm}\|\|^2}$	0.1
	Arm orientation tracking	$e^{-\|\|\sum_j o^j_{arm} - \hat{o}^j_{arm}\|\|^2}$	0.1
	Tray position tracking	$e^{-10\|\|p_{end} - \hat{p}_{end}\|\|^2}$	0.1
	Tray orientation tracking	$e^{-2\|\|o_{end} - \hat{o}_{end}\|\|^2}$	0.05
Task	Object tracking	$e^{-10\|\|p_{obj} - \hat{p}_{obj}\|\|^2}$	0.4
Smoothness	Action rate	$\|\|a_t - a_{t-1}\|\|^2$	-50.0

IV. EXPERIMENT AND RESULTS

A. Experimental Setup

a) Data collection: To pre-train our ConTact framework, we first collect a paired sim-to-real dataset. For the SOT task, we generate diverse 6D pose trajectories by varying the tray's orientation using trigonometric functions with different frequencies. For the COT task, since the composite car has a delicate structure that's easy to collapse, we first train a basic tracking policy using motion capture data for the purpose of safe and efficient data collection, rather than relying on tactile sensing. For both tasks, the control frequency and the tactile sample frequency are set to 50 Hz. The resulting datasets contain 8,000 tactile frames for the SOT task and 10,000 for the COT task, corresponding to 6 minutes of real-world interaction.

b) Simulation Training: We train our policies in 4096 parallel environments simulated in MuJoCo MJX [13]. Each environment features a Kinova Gen3 arm with a tactile tray end-effector. To foster robust sim-to-real transfer, we apply extensive domain randomization at the beginning of each episode. For the SOT task, this includes randomizing the sphere's initial position, size, and density. For the COT task, we randomize the initial relative poses of the car's components. We also vary friction coefficients and add noise to proprioceptive observations across all tasks. An episode terminates if the manipulated object falls off the tray. Key hyperparameters for the training are detailed in TABLE II. The contrastive pre-training of the tactile encoder converges in few minutes, and the downstream RL policy training requires about 8 hours on 2 NVIDIA V100 GPUS.

c) Zero-Shot Real-World Deployment: We deploy the trained policies directly onto a physical Kinova Gen3 arm with a tactile tray, operating at a 50 Hz. The transfer is zero-shot, with the policy network deployed directly without fine-tuning. This is achieved by replacing the simulation encoder with its pre-trained, frozen real-world counterpart, which processes raw tactile signals for the policy.

TABLE II: Key Hyperparameters.

Category	Hyperparameter	Value
Data Pairing	Sliding window size W:	4
	Motion feature pos. weight (α):	0.5
	Motion feature rot. weight (β):	2.0
ConTact	Latent dimension	256
	Contrastive loss weight (λ)	0.1
	Contrastive temperature (τ)	0.5
	Reconstruction weight (w_0, w_Δ)	(20.0, 40.0)
	Batch size	128
DRL	Actor network architecture	[64, 64]
	Critic network architecture	[64, 64, 64]

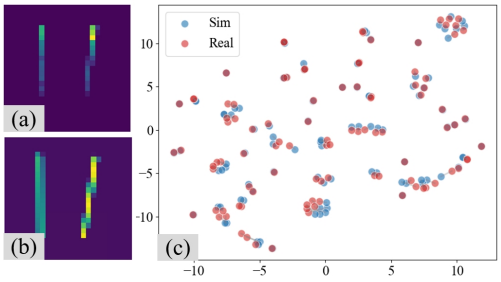


Fig. 5: (a) A real-world tactile image and its paired simulated counterpart (b). (c) t-SNE visualization of their aligned latent features.

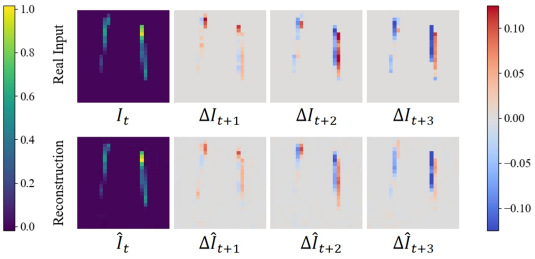


Fig. 6: The real-world tactile input (top row) and the reconstructed tactile signals (bottom row). The first frame is the initial frame, while the subsequent three frames represent frame-to-frame differences.

B. Ablation Study of the Tactile Encoder

We conduct an ablation study to validate our encoder architecture. We design a cross-resolution sim-to-sim alignment task within a simulated environment, pairing tactile data of (25,25) and (50,50) resolutions. As shown in Fig.4, we ablate key components: the temporal processing module (comparing our Conv1D with MLP and LSTM), the spatial modules (CBAM, Res), and the FD. We measure performance using the cosine similarity of paired latent and the average reconstruction loss.

In Fig. 4, our full model ④ achieves the best performance, yielding the highest latent similarity and the lowest reconstruction loss among all tested variants. In the spatial domain, removing the CBAM module ⑤ or residual connections ⑥ degrades performance, which indicates that focusing on salient regions and preserving feature integrity are both critical. In the temporal domain, our lightweight Conv1D network outperforms both MLP-based ① and more complex LSTM-based ② models, as it is simple and sufficient to model the local temporal patterns. Finally, removing the FD strategy ③ also impairs performance, showing the benefit of explicitly encoding temporal dynamics akin to velocity, rather than relying on the network to learn them implicitly.

C. Analysis of Tactile Feature Alignment

We also evaluate the ConTact encoder's ability to bridge the sim-to-real domain gap. A selected pair is shown in Fig. 5(a-b). Fig. 5(c) shows a t-SNE visualization of the learned latent space, where paired tactile sequences from simulation (blue) and the real world (red) are projected. The tight coupling between corresponding points across the entire space demonstrates a strong pairwise alignment. The model thus learns a domain-invariant representation that is robust to the shift from simulation to reality.

To assess the learned representations, we evaluate their ability to reconstruct dynamic tactile sequences from real-world inputs (Fig. 6). The model accurately reconstructs the tactile sequences and the subtle frame-to-frame differences.

D. Compare with Baseline

Binary signals—used by [17] and [18] to simplify tactile information—serve as a baseline for sim-to-real transfer. In Fig. 7, we compare the analysis of this approach, which uncovered fundamental limitations and highlights the necessity of ConTact's feature alignment strategy.

We first evaluated the binary signal on the challenging COT task. Fig. 7 highlights the key difference: our force tactile model provides a pressure map that can detect imbalances, such as one wheel pressing harder (b). In contrast, the binary signal is ambiguous, unable to differentiate this unstable state from a balanced one (c), capturing only the presence of contact. The resulting policy struggles to achieve a high reward compared to our force tactile model (d), because its ambiguous signal cannot detect the imbalances that cause the car-wheel separation.

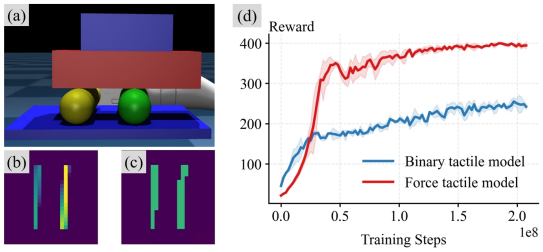
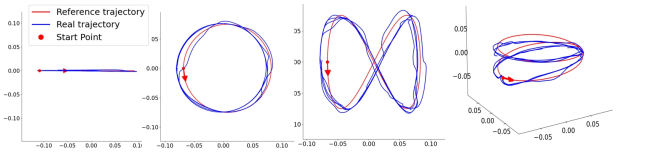


Fig. 7: Comparison of the force-based model and the binary-based model on the COT task (a). The force-based model can detect variations in pressure between the two wheels (as illustrated in figure (b), where deeper colors signify greater pressure), whereas the binary-based model can only identify the wheels' positions (c). Figure (d) illustrates the reward curves, where each curve's shaded region represents the standard deviation of reward values across three different random seeds, indicating the uncertainty in the results.

More critically, even with this simplified binary representation, the sim-to-real gap remains uncrossed. We introduce domain randomization into the simulation (e.g., randomizing the sensor's ray length L) in an attempt to bridge this gap. However, when deployed, the policy is completely unstable, resulting in severe vibrations of the tactile tray and immediate task failure. This demonstrates that for contact-sensitive tasks, even a simple binary signal contains subtle sim-to-real discrepancies that naive randomization cannot overcome. This gap arises because simulated contact often registers as a contiguous patch of signals, whereas real-world contact, due to microscopic surface imperfections, is often sparse and intermittent. These baseline failures, stemming from the ambiguity of simplified signals and an intractable sim-to-real gap, directly motivate our data-driven approach of learning a domain-invariant feature space with ConTact.



(a) Linear Trajectory (b) Circular Trajectory (c) Lemniscate Trajectory (d) Ascending Helical Trajectory

Fig. 8: End-effector trajectory tracking performance in the real world. The plots compare the predefined reference trajectories (red) with the actual trajectories executed by the robot (blue) over five episodes.

E. Real-world Deployment

Our approach enables zero-shot deployment of the learned policy onto a physical Kinova Gen3 arm, and the attainment video shows the stability and robustness of our framework. We evaluate its capability to stabilize objects during dynamic movements across four distinct reference trajectories: linear, circular, lemniscate, and ascending helical. As shown in Fig. 8, the robot's end-effector tracks these diverse paths with fine accuracy, demonstrating robust control.

The policy's effectiveness is shown in challenging COT tasks, as shown in Fig. 9(a-b). The toy car successfully stabilizes along both linear and circular paths. The tactile feedback proves vital for maintaining the car's structural integrity, especially when we test the policy's sensitivity by shifting the car's body forward/backward relative to the wheels at initialization. Fig. 9(a) shows the tactile sensor detecting uneven pressure from the wheels, with the control policy quickly adjusting the tray's orientation to balance the forces and thereby preventing the wheel separation.

To quantify the performance of the zero-shot deployed policy, we analyzed the success rate of the policy under the four reference trajectories. The evaluation criteria for failure are defined as follows: if the manipulated sphere or the composite toy car falls off the tactile tray during the movement, or if the composite toy car in the COT task experiences structural collapse. As shown in III, the policy achieved a 100% success rate in linear and Lemniscate trajectory tests, and an 80% success rate in more complex circular and ascending helical trajectory tests.

TABLE III: Success Rate of ConTact in Different Trajectories. For each trajectory, we conduct 5 independent tests to ensure the reliability of the results. The recovery time is how long it takes to stabilize the object.

Trajectories	Recovery time	Success rate	Failure type
Linear (COT)	4 s	100 %	-
Circular (COT)	7 s	80 %	Collapse
Lemniscate (SOT)	3 s	100 %	-
Ascending Helical (SOT)	5 s	80 %	Drop

Beyond the COT task, the policy demonstrates robustness in SOT by dynamically adjusting the sphere's position across various challenging real-world scenarios(Fig. 9(c)).

Object Generalization: The policy handles spheres of varying diameters (5–8 cm) and materials. In a challenging test, it seamlessly adapts to real-time object switching for over eight consecutive trials without failure (Fig. 9(d)).

Initial Condition Robustness: The policy proves highly robust to starting conditions, consistently recovering from extreme boundary placements with initial offsets up to 13 cm. Fig. 9 (e) depicts successful trials from various distinct off-center positions.

Perturbation Resistance: The policy's robustness is validated against external disturbances. In Fig. 9(f), during a lemniscate trajectory, it recovered and stabilized the object after over 10 consecutive light taps with a hammer.

V. CONCLUSIONS

This paper introduces ConTact, a contrastive learning framework that effectively bridges the sim-to-real gap for tactile-driven robotic manipulation. We also develop a high-speed, ray-casting tactile simulator in MuJoCo MJX and design a spatio-temporal encoder to align features between simulated and real-world tactile sensors. We validate the framework through zero-shot deployment of a DRL controller on a Kinova Gen3. Specifically, our controller successfully executed dynamic balancing tasks (SOT and COT) across four diverse trajectories, demonstrating the efficacy of the sim-to-real tactile features.

Experimental validation relies on a single tactile sensor, inevitably limiting the ConTact framework's generalizability across different tactile hardware. Meanwhile, our manipulation tasks are also restricted to tray-based object stabilization, leaving complex dynamics such as sliding contacts and slip detection to be fully addressed. In the future, we will deploy our ConTact on more diverse types of tactile sensors in real-world to verify its versatility across various platforms. Additionally, we will design more contact-rich tasks using robots equipped with diverse dexterous hands, further expanding the scope of tactile-driven control.

REFERENCES

[1] L. Han, Q. Zhu, J. Sheng, C. Zhang, T. Li, Y. Zhang, H. Zhang, Y. Liu, C. Zhou, R. Zhao *et al.*, "Lifelike agility and play in quadrupedal robots using reinforcement learning and generative pre-trained models," *Nature Machine Intelligence*, vol. 6, no. 7, pp. 787–798, 2024.

[2] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 674–691, 2021.

[3] S. Kadalagere Sampath, N. Wang, H. Wu, and C. Yang, "Review on human-like robot manipulation using dexterous hands," *Cognitive Computation and Systems*, vol. 5, no. 1, pp. 14–29, 2023.

[4] T. Tsuji, Y. Kato, G. Solak, H. Zhang, T. Petrič, F. Nori, and A. Ajoudani, "A survey on imitation learning for contact-rich tasks in robotics," *arXiv preprint arXiv:2506.13498*, 2025.

[5] B. He, Q. Miao, Y. Zhou, Z. Wang, G. Li, and S. Xu, "Review of bioinspired vision-tactile fusion perception (vtfp): From humans to humanoids," *IEEE Transactions on Medical Robotics and Bionics*, vol. 4, no. 4, pp. 875–888, 2022.

[6] C. Ciliberto, L. Fiorio, M. Maggiali, L. Natale, L. Rosasco, G. Metta, G. Sandini, and F. Nori, "Exploiting global force torque measurements for local compliance estimation in tactile arrays," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 3994–3999.

[7] M. Meribout, N. A. Takele, O. Derege, N. Rifiki, M. El Khalil, V. Tiwari, and J. Zhong, "Tactile sensors: A review," *Measurement*, p. 115332, 2024.

[8] J. Tegin and J. Wikander, "Tactile sensing in intelligent robotic manipulation–a review," *Industrial Robot: An International Journal*, vol. 32, no. 1, pp. 64–70, 2005.

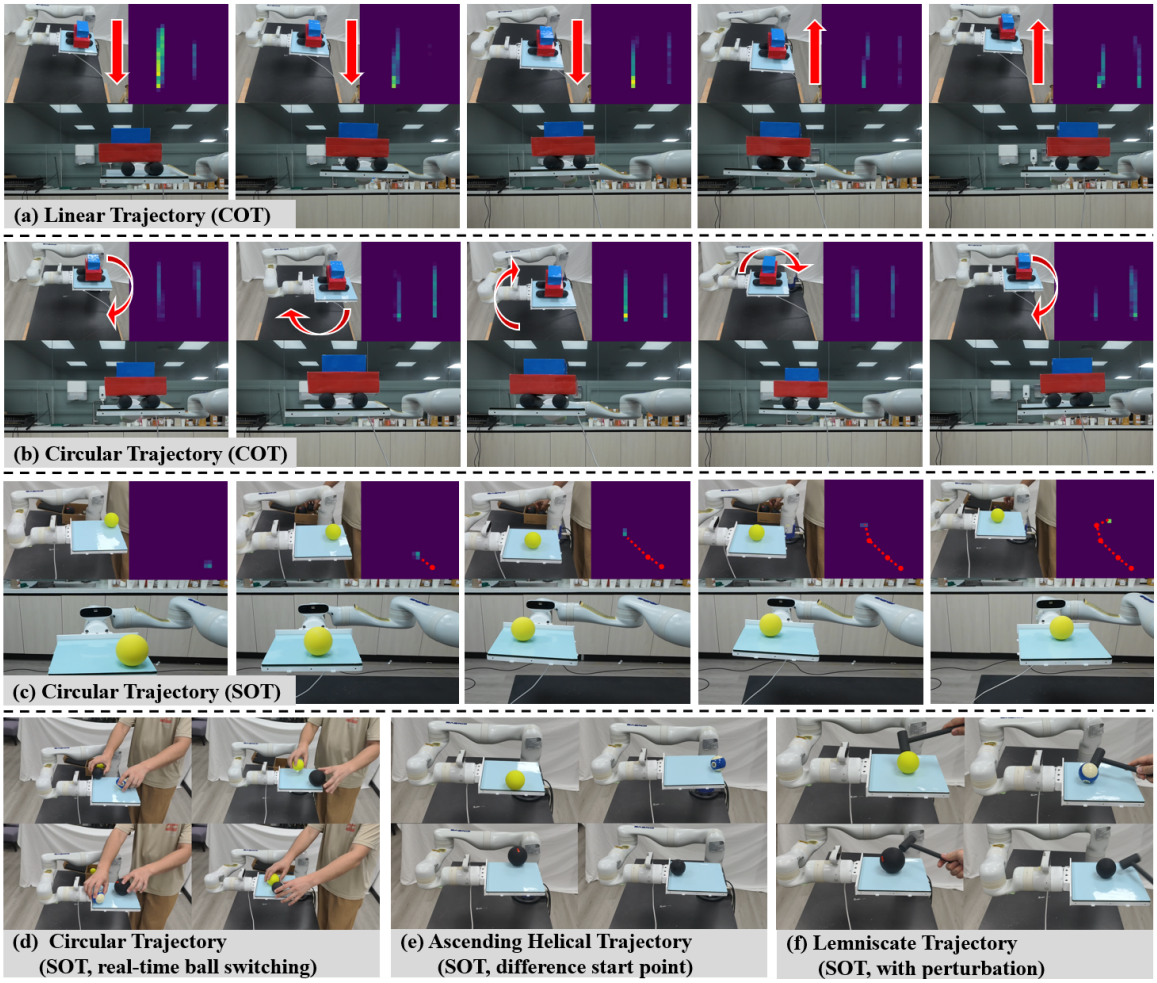


Fig. 9: Visual results from the zero-shot real-world policy deployment. (a, b) The policy stabilizes the composite car in the COT task along linear and circular trajectories. (c) Successful stabilization of a sphere in the SOT task. (d-f) The policy demonstrates robustness against dynamic challenges, including real-time object switching, varied initial positions, and external perturbations.

[9] I. Akinola, J. Xu, J. Carius, D. Fox, and Y. Narang, "Tacsl: A library for visuotactile sensor simulation and learning," *IEEE Transactions on Robotics*, vol. 41, pp. 2645–2661, 2025.

[10] S. Wang, M. Lambeta, P.-W. Chou, and R. Calandra, "Tacto: A fast, flexible, and open-source simulator for high-resolution vision-based tactile sensors," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3930–3937, 2022.

[11] Z. Si and W. Yuan, "Taxim: An example-based simulation model for gelsight tactile sensors," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2361–2368, 2022.

[12] Y. Li, W. Du, C. Yu, P. Li, Z. Zhao, T. Liu, C. Jiang, Y. Zhu, and S. Huang, "Taccel: Scaling up vision-based tactile robotics via high-performance gpu simulation," *arXiv preprint arXiv:2504.12908*, 2025.

[13] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.

[14] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[15] M. Mittal, P. Roth, J. Tigue, A. Richard, O. Zhang, P. Du, A. Serrano-Munoz, X. Yao, R. Zurbrügg, N. Rudin *et al.*, "Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning," *arXiv preprint arXiv:2511.04831*, 2025.

[16] Z. Ding, Y.-Y. Tsai, W. W. Lee, and B. Huang, "Sim-to-real transfer for robotic manipulation with tactile sensory," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 6778–6785.

[17] Z. Ding, N. F. Lepora, and E. Johns, "Sim-to-real transfer for optical tactile sensing," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1639–1645.

[18] P. Ruppel, Y. Jonetzko, M. Görner, N. Hendrich, and J. Zhang, "Simulation of the syntouch biotac sensor," in *Intelligent Autonomous Systems 15: Proceedings of the 15th International Conference IAS-15*. Springer, 2019, pp. 374–387.

[19] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PmLR, 2021, pp. 8748–8763.

[20] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[21] X. B. Peng, P. Abbeel, S. Levine, and M. Van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Transactions On Graphics (TOG)*, vol. 37, no. 4, pp. 1–14, 2018.

[22] T. He, Z. Luo, X. He, W. Xiao, C. Zhang, W. Zhang, K. M. Kitani, C. Liu, and G. Shi, "Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning," in *Conference on Robot Learning*. PMLR, 2025, pp. 1516–1540.

[23] X. Cheng, J. Li, S. Yang, G. Yang, and X. Wang, "Open-television: Teleoperation with immersive active visual feedback," in *Conference on Robot Learning*. PMLR, 2025, pp. 2729–2749.

[24] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.