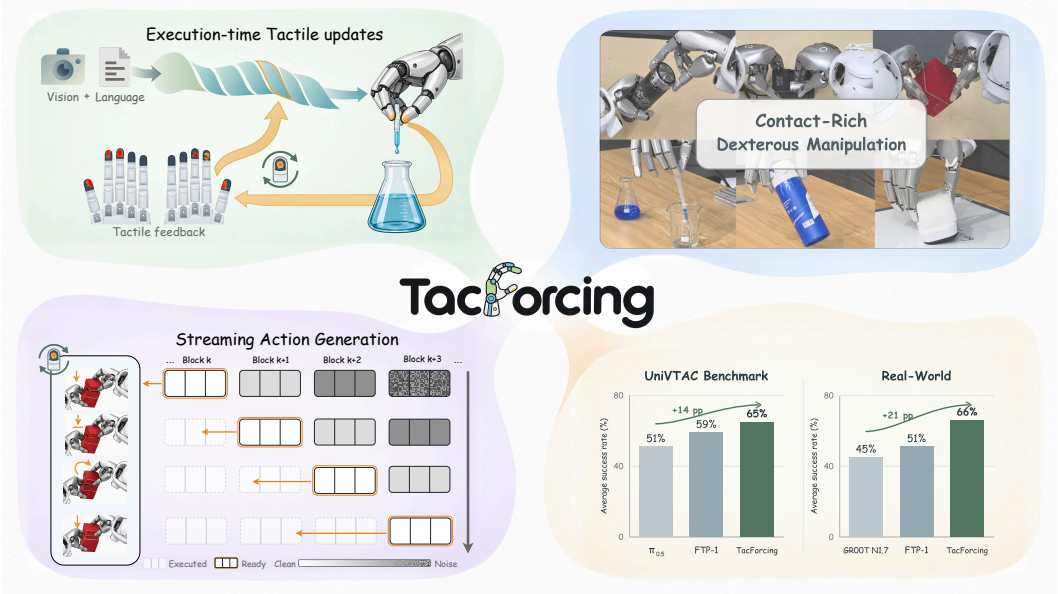
TACFORCING: STREAMING ACTION GENERATION WITH EXECUTION-TIME TACTILE FEEDBACK

Anonymous authors
Paper under double-blind review



ABSTRACT

Contact-rich manipulation requires adapting to contact states that can evolve substantially within an action horizon. However, chunk-based vision-language-action models predict complete action chunks from observations collected before execution, leaving tactile conditioning stale during execution. Existing tactile-reactive approaches typically rely on separate high-frequency controllers, which increase both architectural and training complexity. In this paper, we introduce **TacForcing**, a streaming action-generation framework incorporating execution-time tactile feedback. TacForcing replaces the standard action expert with a streaming expert that generates action blocks sequentially while preserving intermediate states of unfinished blocks. After each block is executed, the expert resumes generation from these states using newly acquired tactile feedback. To better align tactile conditioning with action execution, we further introduce Execution-Aware Tactile Attention (EATA), which restricts direct access to each tactile update to the next block scheduled for execution. Across six UniVTAC simulation tasks and six real-world contact-rich manipulation tasks on two robot platforms, TacForcing achieves average success rates of 65% in simulation and 66% in the real world, outperforming the strongest baselines by 6 and 15 percentage points, respectively.
Project Website: https://tacforcing-review.pages.dev/

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

Contact-rich tasks, including precision assembly, part insertion, and dexterous manipulation, require both semantic understanding and continuous adaptation to evolving contact states (Huang et al., 2024; Zhou et al., 2026). Vision-language-action (VLA) models generate robot actions from language instructions and visual observations and have demonstrated broad generalization across manipulation tasks (Black et al., 2024; Bjorck et al., 2025). However, visual observations alone cannot reliably reveal changes in force, slip events, or other latent contact states, particularly under occlusion (Zhou et al., 2026). This perceptual limitation reduces the precision and robustness of contact-rich manipulation.

Recent studies have integrated tactile perception into VLA models for contact-rich manipulation (Huang et al., 2025; Cheng et al., 2026; Yuan et al., 2026). However, as illustrated in Figure 2, tactile representations can change substantially within a single action chunk even when visual representations remain nearly unchanged. Consequently, conditioning an entire action chunk on a fixed tactile observation creates a growing temporal mismatch between the tactile input and the contact states encountered during execution (Xue et al., 2025). Existing reactive methods mitigate this mismatch by coupling chunk-level action prediction with dedicated high-frequency tactile pathways (Xue et al., 2025; Li et al., 2026; Niu et al., 2026; Zhang et al., 2026). These pathways enable execution-time adjustments based on newly acquired tactile feedback. However, the additional policy-specific components increase architectural and training complexity, motivating a more integrated approach that incorporates execution-time tactile feedback directly into action generation.

We therefore introduce TacForcing (Figure 3), a streaming action-generation framework that effectively incorporates execution-time tactile feedback. To adapt to evolving contact states, TacForcing partitions each action chunk into sequential blocks and employs a Streaming Action Expert to generate these blocks progressively during execution. Upon completion, each block is dispatched for execution, while the intermediate states of all unfinished blocks are retained for subsequent refinement. Generation then resumes from these states using newly acquired tactile feedback. However, streaming generation jointly refines all unfinished blocks, allowing later blocks to directly attend to tactile feedback that may become outdated before their execution. To address this mismatch, we introduce Execution-Aware Tactile Attention (EATA), which restricts direct access to each tactile update to the next block scheduled for execution.

We evaluate TacForcing on six tasks from the UniVTAC simulation benchmark (Chen et al., 2026) and six real-world contact-rich manipulation tasks across two robot platforms, comparing against representative vision-only, tactile-conditioned, and tactile-reactive policies. TacForcing achieves average success rates of 65% in simulation and 66% in the real world, exceeding the strongest baselines by 6 percentage points in simulation and 15 percentage points in the real world. Ablation studies further show that both execution-time tactile updates and EATA contribute to improved task success. In summary, our contributions are as follows:

- We introduce TacForcing, a streaming action-generation framework that effectively incorporates execution-time tactile feedback without a separate reactive controller.
- We adapt a Streaming Action Expert to incorporate execution-time tactile feedback and introduce Execution-Aware Tactile Attention (EATA), which reduces temporal mismatch by restricting direct access to each tactile update to the next block to execute.
- Experiments demonstrate that TacForcing improves contact-rich manipulation performance across diverse simulated and real-world tasks.

2 PRELIMINARIES AND MOTIVATION

2.1 FLOW MATCHING

Flow Matching (FM) (Lipman et al., 2022) learns a conditional velocity field that transports samples from a Gaussian prior to the data distribution. Given condition y, we sample $x_1 \sim p_{\text{data}}(\cdot \mid y)$, $x_0 \sim \mathcal{N}(0, I)$, and $\tau \sim \mathcal{U}(0, 1)$, and define the linear interpolation path and target velocity as:

$$x^\tau = (1 - \tau)x_0 + \tau x_1, \qquad u^\star = x_1 - x_0. \tag{1}$$

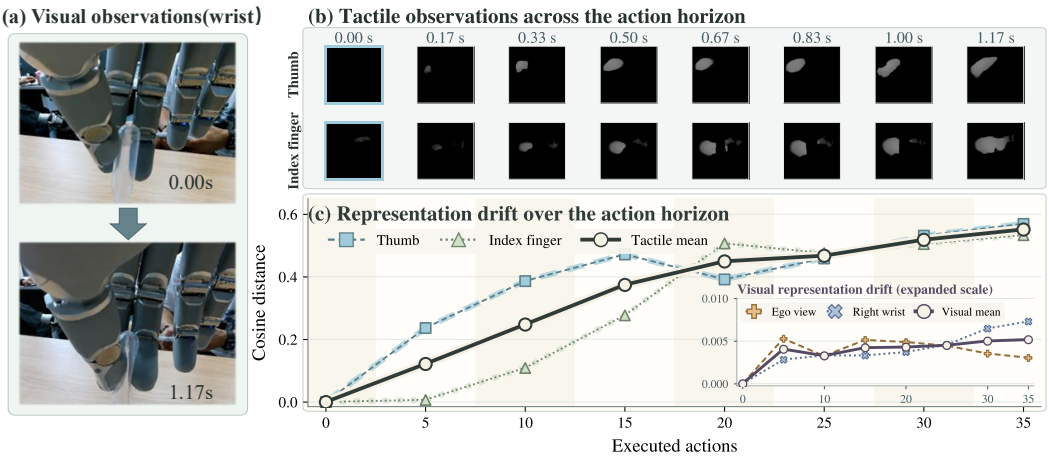


Figure 2: Visual and tactile dynamics during dropper squeezing. (a) Visual observations at the beginning and end of the action horizon. (b) Deformation maps from the thumb and index finger, sampled every five actions. (c) Cosine distances from the initial visual and tactile representations; the inset provides an enlarged view of the visual cosine distances.

The model is trained by minimizing

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{y,x_1,x_0,\tau}\left[\|v_\theta(x^\tau,\tau,y) - u^\star\|_2^2\right]. \tag{2}$$

During inference, integrating the learned velocity field v_θ from $\tau = 0$ to $\tau = 1$ transforms a sample from the Gaussian prior into a sample from the conditional data distribution.

2.2 Problem Formulation

We consider language-conditioned, contact-rich robot manipulation. At decision step t, the robot receives a visual observation V_t, a proprioceptive state s_t, a tactile observation T_t, and a language instruction ℓ. Let $c_t = (V_t, s_t, \ell)$ denote the task context. Conditioned on (c_t, T_t), a conventional chunk-based policy models $p_\theta(A_t \mid c_t, T_t)$ and generates an action chunk with horizon H,

$$A_t = (a_t, a_{t+1}, \ldots, a_{t+H-1}), \qquad a_{t+i} \in \mathbb{R}^{d_a}, \tag{3}$$

where d_a denotes the action dimension. In a conventional flow-based action expert, the action chunk A_t and Gaussian noise ϵ of the same shape serve as the data and prior endpoints, respectively. All action positions share a common flow time and are generated synchronously. The entire chunk is generated from fixed initial inputs (c_t, T_t) before execution begins, so tactile feedback acquired during execution does not update the remaining actions within that chunk.

2.3 Temporal Mismatch from Fixed Tactile Conditioning

To characterize how visual and tactile observations evolve within an action chunk, we analyze a representative dropper-squeezing episode (Figure 2). Over a 40-action horizon, we sample both modalities every five control steps, yielding eight observations from step 0 to step 35 (1.17 s after the initial observation). For each modality, we average distances across its two sensors: the top and wrist cameras for vision, and the thumb and index finger for touch. At step 35, the sensor-averaged cosine distance from the initial representation is approximately 0.005 for vision and 0.55 for touch, indicating substantial changes in tactile representations while visual representations remain nearly unchanged in this episode. Details of representation extraction and distance computation are provided in Appendix A.3. This example illustrates that tactile representations can change substantially within an action chunk despite limited changes in visual representations, motivating the use of execution-time tactile feedback rather than relying solely on the initial observation.

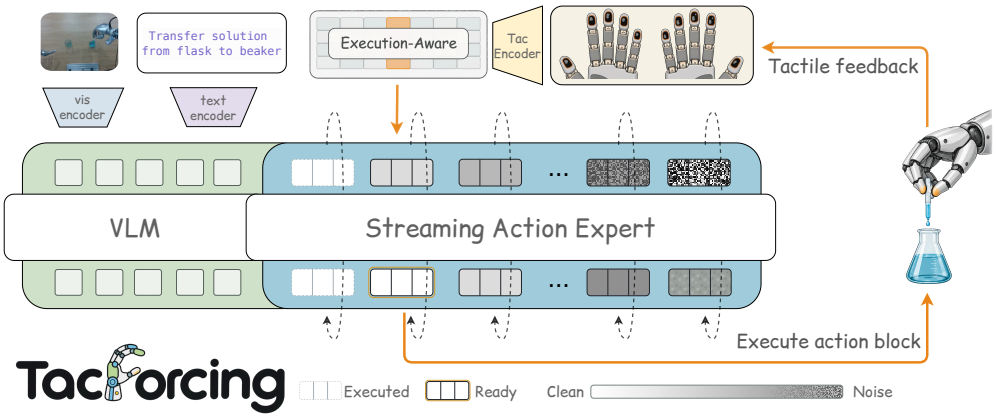


Figure 3: **Overview of TacForcing**. The VLM encodes the visual observation and language instruction once, and the resulting task context is reused throughout streaming generation. The Streaming Action Expert progressively refines action blocks according to block-specific flow times, allowing successive blocks to become ready for sequential execution. After each ready block is executed, the resulting tactile feedback is encoded before generation resumes from the retained intermediate states of unfinished blocks. Execution-Aware Tactile Attention restricts direct access to the latest tactile feedback to the block scheduled for execution next, thereby reducing the temporal mismatch between tactile acquisition and action execution.

3 METHOD

In this section, we present **TacForcing** (Figure 3), a streaming action-generation framework that effectively incorporates execution-time tactile feedback. We first introduce block-wise streaming action generation in Section 3.1 and then describe execution-time tactile conditioning in Section 3.2. Finally, we describe the training procedure in Section 3.3.

3.1 STREAMING ACTION GENERATION VIA BLOCK-WISE FLOW SCHEDULING

Streaming Action Expert. As discussed in Section 2.2, a conventional flow-based Action Expert generates the entire action chunk synchronously using only observations acquired before execution. Consequently, tactile feedback obtained during execution cannot condition the actions that remain to be executed. TacForcing addresses this limitation by replacing the standard expert with a Streaming Action Expert that generates the action chunk progressively. Near-term actions are completed and dispatched for execution first, while the intermediate states of later actions are retained for further refinement. This design enables newly acquired tactile feedback to condition the remaining actions within the same chunk without requiring a separate reactive controller.

Block-wise flow scheduling. Let N denote the total number of sampling steps, with $n \in \{1, \dots, N\}$. Building on prior work in streaming generation (Chen et al., 2024; Lu et al., 2026), we replace the flow time shared across an entire action chunk with position-dependent flow times. At sampling step n, we represent these flow times as

$$\boldsymbol{\tau}^{(n)} = \left(\tau_1^{(n)}, \tau_2^{(n)}, \dots, \tau_H^{(n)}\right) \in [0,1]^H, \tag{4}$$

where $\tau_i^{(n)}$ denotes the flow time of the i-th action position. We coordinate action completion and tactile updates at the block level, incorporating new tactile feedback after each block is executed.

Assuming $H = KB$ and $N = KS$, we partition the action chunk into K consecutive blocks of B actions and separate successive block completions by S sampling steps. Here, B determines the tactile update interval in executed actions, while S counts numerical integration steps between

successive block completions. Let $A_t^{(k)}$ denote the k-th block. The function

$$b(i) = \left\lfloor \frac{i-1}{B} \right\rfloor + 1$$

maps action position i to its corresponding block. All actions in block k share a block flow time λ_k and reach completion simultaneously at sampling step $n_k = kS$. The flow-time trajectory is

$$\lambda_k^{(n)} = \min\left(\frac{n}{n_k}, 1\right), \qquad k \in \{1, \ldots, K\}. \tag{5}$$

Accordingly, the flow time of action position i is $\tau_i^{(n)} = \lambda_{b(i)}^{(n)}$. At each sampling step, we use explicit Euler integration to advance every unfinished block, with a block-specific step size $\Delta\lambda_k^{(n)} = \lambda_k^{(n)} - \lambda_k^{(n-1)}$ determined by Equation 5. Completed blocks remain fixed because their step sizes are zero. Since $n_1 < n_2 < \cdots < n_K$, the blocks become ready for execution sequentially. When $A_t^{(k)}$ completes at n_k, it becomes available for execution, while later blocks remain partially generated. Their intermediate states are retained and further refined after $A_t^{(k)}$ is executed.

3.2 Conditioning on Execution-Time Tactile Feedback

Block-level tactile updates. Within each action chunk, the task context c_t is encoded once as $C_t = f_{\text{ctx}}(c_t)$ and reused throughout chunk generation. In contrast, the tactile condition is refreshed after the execution of each block. Let $T_t^{(k)}$ denote the tactile observation acquired after the first k blocks have been executed, with $T_t^{(0)} = T_t$. The deformation maps from the M fingertips are encoded independently using a shared tactile encoder f_{tac}, producing the tactile tokens

$$Z_t^{(k)} = \left(z_{t,1}^{(k)}, \ldots, z_{t,M}^{(k)}\right).$$

During sampling steps $(k-1)S < n \leq kS$, $Z_t^{(k-1)}$ serves as the latest available tactile representation. After $A_t^{(k)}$ is executed, the newly acquired observation $T_t^{(k)}$ is encoded as $Z_t^{(k)}$ and used to condition the subsequent refinement of the remaining blocks.

Execution-Aware Tactile Attention. After the first $k-1$ blocks have been executed, $Z_t^{(k-1)}$ represents the latest contact state and is temporally aligned with $A_t^{(k)}$, the block scheduled to execute next. However, it should not directly condition every unfinished block. Later blocks will be executed only after additional tactile updates and, because contact states can change rapidly as illustrated in Figure 2, may encounter states that differ substantially from the state encoded by $Z_t^{(k-1)}$.

Despite this temporal distinction, the Streaming Action Expert advances all unfinished blocks jointly. Without an additional constraint, unrestricted attention would allow every unfinished action token to attend directly to $Z_t^{(k-1)}$, thereby conditioning later blocks on tactile feedback that may become outdated before their execution. We therefore introduce EATA to restrict direct access to $Z_t^{(k-1)}$ to the action tokens in $A_t^{(k)}$. We implement this restriction by applying an additive mask to the action-to-tactile attention scores, while preserving the expert's remaining attention structure:

$$\mathcal{M}_{i,m}^{(k)} = \begin{cases} 0, & b(i) = k, \\ -\infty, & \text{otherwise}, \end{cases} \tag{6}$$

where $i \in \{1, \ldots, H\}$ indexes action queries and $m \in \{1, \ldots, M\}$ indexes tactile keys. Under this mask, the current tactile representation directly conditions only the block scheduled to execute next. Later blocks continue to evolve according to the block-wise flow schedule without directly accessing the current tactile tokens and gain direct access to updated tactile feedback when they become the next block to execute. We enforce the same constraint on direct access during training to ensure consistency with the temporal conditioning used during streaming inference.

3.3 TRAINING

To match streaming inference, we train the expert on block-wise intermediate states paired with the tactile feedback available at the corresponding execution stage. For each demonstrated action chunk A_t, we sample Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$ and a normalized generation progress $p \sim \mathcal{U}([0, 1))$. Using $n = pN$ and $N = KS$ in Equation 5 yields the flow time and interpolated state of block k as

$$\lambda_k(p) = \min\left(\frac{Kp}{k}, 1\right), \qquad \widetilde{A}_t^{(k)} = \big(1 - \lambda_k(p)\big)\epsilon^{(k)} + \lambda_k(p) A_t^{(k)}. \tag{7}$$

The block states form $\widetilde{A}_t$, with action-wise flow times $\tau_i(p) = \lambda_{b(i)}(p)$ collected in $\boldsymbol{\tau}(p)$.

For the sampled progress p, let $k^\star(p) = \lfloor Kp \rfloor + 1$. We use the corresponding tactile representation $Z_t^{(k^\star-1)}$ and EATA mask $\mathcal{M}^{(k^\star)}$. In each demonstration, the selected tactile frame corresponds to offset $(k^\star - 1)B$ from the chunk start and precedes execution of block $k^\star$. Let

$$\mathcal{U}(p) = \big\{ i \in \{1, \dots, H\} \mid \lambda_{b(i)}(p) < 1 \big\}$$

denote the unfinished action positions. For each $i \in \mathcal{U}(p)$, the predicted velocity is

$$\widehat{v}_{\theta,i}(p) = v_{\theta,i}\Big(\widetilde{A}_t, \boldsymbol{\tau}(p), C_t, Z_t^{(k^\star-1)}; \mathcal{M}^{(k^\star)}\Big).$$

The training objective is

$$\mathcal{L}_{\text{train}} = \mathbb{E}_{A_t,\epsilon,p}\left[\frac{1}{|\mathcal{U}(p)|}\sum_{i\in\mathcal{U}(p)} \|\widehat{v}_{\theta,i}(p) - (a_{t+i-1} - \epsilon_i)\|_2^2\right]. \tag{8}$$

The loss is applied only to unfinished actions, matching the positions updated at the corresponding stage of streaming inference.

4 EXPERIMENTS

We first describe the experimental setup, including the benchmarks, baselines, and implementation details, in Section 4.1. We then present the main quantitative results from the simulation benchmark and the real-world platforms in Section 4.2. We assess the effects of execution-time tactile conditioning and Execution-Aware Tactile Attention through ablation studies in Section 4.3.

4.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate TacForcing on six contact-rich manipulation tasks from the UniVTAC simulation benchmark (Chen et al., 2026) and six real-world tasks across two robot platforms. The real-world tasks comprise *Close Cap*, *Insert Battery*, and *Close Lid* on Sharpa North, and *Transfer Liquid*, *Wipe Board*, and *Stand Bottle* on Sharpa & RealMan (Figure 4). Details of the simulation tasks and real-world platforms and tasks are provided in Appendices C and B, respectively.

Baselines. We compare TacForcing against representative vision-only, tactile-conditioned, and tactile-reactive policies. In simulation, we evaluate four baselines: the vision-only $\pi_{0.5}$ (Intelligence et al., 2025); UniVTAC-ACT (Chen et al., 2026), which augments an Action Chunking Transformer with a pretrained tactile encoder; FTP-1 (Yuan et al., 2026), which maps heterogeneous tactile observations to unified tokens processed by a shared tactile expert; and RDP (Xue et al., 2025), which combines low-frequency action-chunk prediction with a high-frequency tactile-reactive controller. We use the released checkpoints of UniVTAC-ACT and FTP-1 and train $\pi_{0.5}$ and RDP using their official implementations. In the real world, we evaluate $\pi_{0.5}$ and GR00T N1.7 (Bjorck et al., 2025) as vision-only baselines, FTP-1 as a tactile-conditioned baseline, and T-Rex (Niu et al., 2026) as a tactile-reactive baseline, all trained using their official implementations. Within each setting, all methods follow the same evaluation protocol.

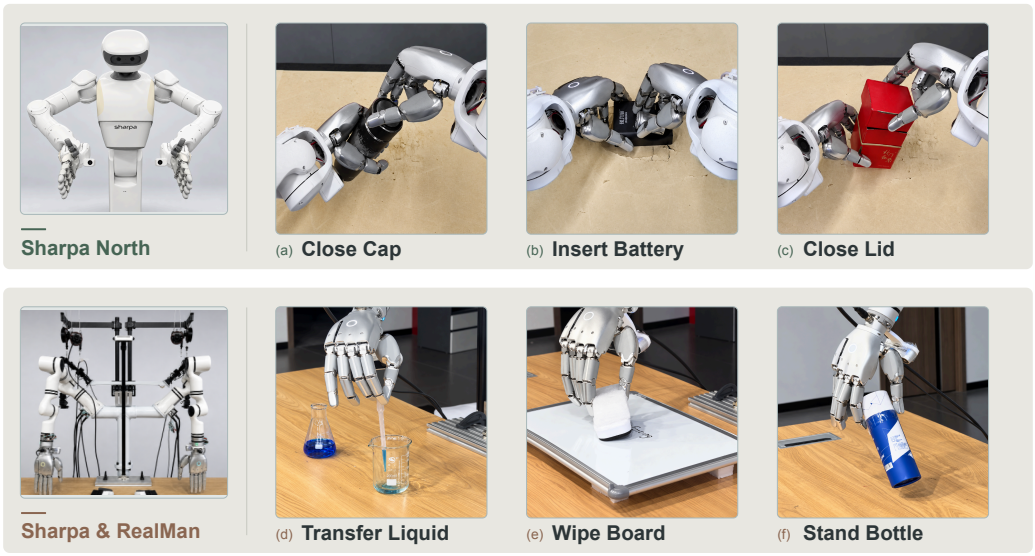


Figure 4: **Real-world platforms and tasks.** Top: *Close Cap*, *Insert Battery*, and *Close Lid* on Sharpa North. Bottom: *Transfer Liquid*, *Wipe Board*, and *Stand Bottle* on Sharpa & RealMan. Platform views are shown on the left; task scenes are shown on the right.

Implementation Details. TacForcing is initialized from $\pi_{0.5}$ (Intelligence et al., 2025) in simulation and GR00T N1.7 (Bjorck et al., 2025) in the real world. In both settings, the tactile encoder is initialized from the pretrained tactile encoder of FTP-1 (Yuan et al., 2026). We use a block size of $B = 5$. In simulation, an action horizon of $H = 50$ is divided into $K = 10$ blocks. In the real-world setting, we use $H = 40$, $K = 8$. We report success rates over 100 rollouts per simulation task and 16 independent trials per real-world task. Additional details are provided in Appendix A.

4.2 Main Results

Table 1: **Results on the UniVTAC simulation benchmark.** Success rates (%). The highest and second-highest distinct values in each column are shown in **bold** and underlined, respectively.

Method	Lift Bottle	Pull-out Key	Lift Can	Put Bottle in Shelf	Insert Hole	Insert Tube	Avg.
$\pi_{0.5}$	88	43	46	**43**	39	48	51
UniVTAC-ACT	59	41	24	6	36	58	37
RDP	84	18	12	41	23	75	42
FTP-1	89	35	**66**	23	62	76	59
TacForcing (Ours)	**90**	**48**	63	**43**	**69**	**79**	**65**

Simulation Results. As shown in Table 1, TacForcing achieves the highest average success rate of 65%, outperforming FTP-1 at 59%, the vision-only $\pi_{0.5}$ baseline at 51%, and the tactile-reactive RDP baseline at 42%. At the task level, TacForcing ranks first or ties for first on five of the six tasks. On *Insert Hole* and *Insert Tube*, it improves over the strongest baseline, FTP-1, by 7 and 3 percentage points, respectively. On *Lift Can*, it achieves 63% success, slightly below FTP-1 at 66%. These results support TacForcing's effectiveness across the evaluated contact-rich tasks.

Table 2: **Results on six real-world tasks.** All values are success rates (%). The highest and second-highest distinct rates in each column are shown in **bold** and underlined, respectively.

Method	Sharpa North			Sharpa & RealMan			Avg.
	Close Cap	Insert Battery	Close Lid	Transfer Liquid	Wipe Board	Stand Bottle	
$\pi_{0.5}$	25	38	13	6	31	44	26
GR00T N1.7	44	75	25	19	50	56	45
FTP-1	50	63	38	6	**75**	75	51
T-Rex	44	50	44	31	63	75	51
TacForcing (Ours)	**56**	**81**	**50**	**50**	**75**	**81**	**66**

Real-World Results. As shown in Table 2, TacForcing achieves the highest average success rate of 66%, outperforming FTP-1 and T-Rex at 51% and its vision-only backbone, GR00T N1.7, at 45%. At the task level, it ranks first on five tasks and ties with FTP-1 on *Wipe Board*. On Sharpa North, TacForcing exceeds the best baseline by 6 percentage points on each of the three tasks. On Sharpa & RealMan, its largest gain occurs on *Transfer Liquid*, with 50% success compared with 31% for T-Rex; it also improves *Stand Bottle* success to 81%, compared with 75% for FTP-1 and T-Rex. Overall, these results support the effectiveness of TacForcing across both robot platforms.

4.3 Ablation Study

We evaluate execution-time tactile updates and EATA using three variants with the same streaming generation scheme: (i) fixed tactile input without EATA, (ii) tactile updates without EATA, and (iii) full TacForcing. Comparing (i) with (ii) evaluates the benefit of refreshing tactile observations during chunk execution, while comparing (ii) with (iii) assesses the additional contribution of EATA. Figure 5 reports results on three simulation tasks and three real-world tasks on Sharpa & RealMan.

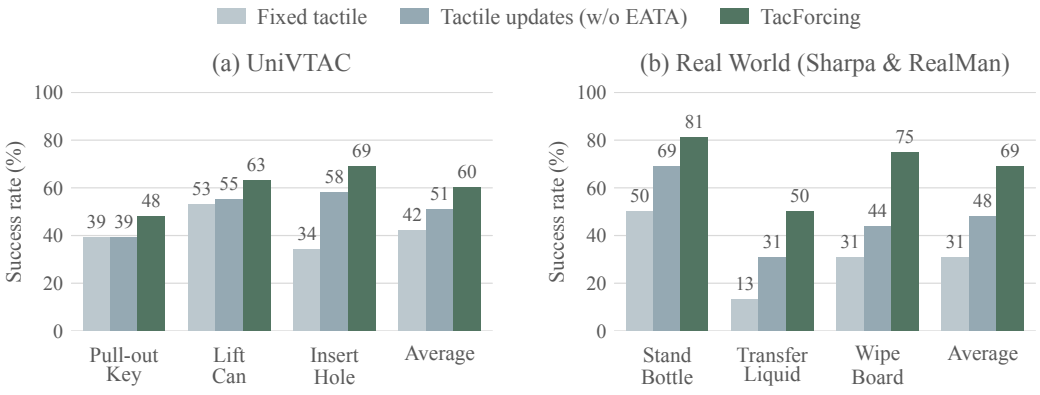


Figure 5: **Ablation results.** All variants retain streaming generation. Fixed tactile holds the initial tactile observation constant within each action chunk, while tactile updates refresh the observation during execution; both disable EATA. TacForcing enables both updates and EATA. Average denotes the mean success rate over the three tasks in each panel.

Effect of Execution-Time Tactile Updates. With EATA disabled, refreshing tactile observations increases the average success rate from 42% to 51% in simulation and from 31% to 48% in the real world. Compared with fixed tactile input, updates improve *Lift Can* and *Insert Hole* while leaving *Pull-out Key* unchanged, and improve all three real-world tasks. The largest simulation gain occurs on *Insert Hole*, where the success rate increases from 34% to 58%. These comparisons support using updated tactile feedback during streaming generation rather than relying on fixed input.

Effect of Execution-Aware Tactile Attention. With execution-time tactile updates enabled, adding EATA increases the average success rate from 51% to 60% in simulation and from 48% to 69% in the real world. Success rates increase on all six tasks, with the largest real-world gain on *Wipe Board*, where the success rate rises from 44% to 75%. These results highlight the importance of temporal alignment between tactile conditioning and action execution.

5 Related Work

Diffusion Models for Sequence Generation. Diffusion and flow-based policies generate action chunks through iterative sampling (Chi et al., 2025; Lipman et al., 2022; Black et al., 2024). Position-dependent denoising supports sequential generation in text, motion, and video (Wu et al., 2023; Zhang et al., 2024; Ruhe et al., 2024). Diffusion Forcing (Chen et al., 2024) assigns independent noise levels to sequence tokens, while streaming policies refine future actions as new observations arrive (Høeg et al., 2025; Chen et al., 2025). FASTER (Lu et al., 2026) uses a horizon-aware flow schedule to complete near-term actions earlier and emit them progressively. TacForcing builds on this streaming perspective to incorporate execution-time tactile feedback into retained intermediate action states within the same action chunk.

Tactile-Aware Policies for Contact-Rich Manipulation. Tactile representation learning provides transferable features across sensors and tasks (Zhao et al., 2024). Sparsh (Higuera et al., 2024) learns general-purpose tactile features through self-supervised pretraining, while AnyTouch (Feng et al., 2025) combines static and dynamic tactile information to support transfer across sensors. Tactile-aware VLAs integrate contact signals into action generation (Huang et al., 2025; Bi et al., 2026; Yuan et al., 2026). Reactive methods combine slower visuomotor planning with faster tactile updates (Li et al., 2026; Niu et al., 2026). RDP (Xue et al., 2025) implements this separation through a slow latent diffusion policy and a fast asymmetric tokenizer for tactile feedback control. Predictive and predictive-reactive approaches model future contact states to guide action generation or online refinement (Zang et al., 2026; Zhou et al., 2026; Zhang et al., 2026). TacForcing refines unfinished action blocks using observed tactile feedback, with EATA restricting direct access to each update to the next block scheduled for execution. It requires neither a separate reactive controller nor explicit tactile prediction to adapt actions during chunk execution.

6 Conclusion

In this work, we introduced TacForcing, a streaming action-generation framework that effectively incorporates execution-time tactile feedback without a separate reactive controller. TacForcing employs a Streaming Action Expert to generate action blocks progressively during execution, retaining the intermediate states of unfinished blocks and refining them using newly acquired tactile feedback. Execution-Aware Tactile Attention (EATA) restricts direct access to each tactile update to the next block scheduled for execution, thereby reducing the temporal mismatch between tactile acquisition and action execution. Across six simulated UniVTAC tasks and six real-world contact-rich manipulation tasks, TacForcing achieves average success rates of 65% and 66%, respectively, outperforming strong baselines in both settings. Ablation results show that both execution-time tactile updates and EATA improve performance, highlighting the value of aligning tactile conditioning with block execution for contact-rich manipulation. The current implementation uses fixed block boundaries and retains visual and proprioceptive inputs within each chunk. Future work could explore adaptive block lengths and the joint scheduling of visual, proprioceptive, and tactile updates.

AI USE STATEMENT

We used generative AI tools to assist with manuscript editing and polishing, identifying and retrieving relevant literature, and supporting research ideation and execution through methodological discussions, code inspection, and feedback on the interpretation of experimental results. The authors reviewed all AI-assisted content and take full responsibility for the final manuscript.

REFERENCES

Jianxin Bi, Kevin Yuchen Ma, Ce Hao, Mike Zheng Shou, and Harold Soh. Vla-touch: Enhancing vision-language-action model with dual-level tactile feedback. *IEEE Robotics Autom. Lett.*, 11(7): 8487–8494, 2026. doi: 10.1109/LRA.2026.3692345. URL https://doi.org/10.1109/LRA.2026.3692345.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith LLontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: an open foundation model for generalist humanoid robots. *CoRR*, abs/2503.14734, 2025. doi: 10.48550/ARXIV.2503.14734. URL https://doi.org/10.48550/arXiv.2503.14734.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. *CoRR*, abs/2410.24164, 2024. doi: 10.48550/ARXIV.2410.24164. URL https://doi.org/10.48550/arXiv.2410.24164.

Baijun Chen, Weijie Wan, Tianxing Chen, Xianda Guo, Congsheng Xu, Yuanyang Qi, Haojie Zhang, Longyan Wu, Tianling Xu, Zixuan Li, Yizhe Wu, Rui Li, Xiaokang Yang, Ping Luo, Wei Sui, and Yao Mu. Univtac: A unified simulation platform for visuo-tactile manipulation data generation, learning, and benchmarking. *CoRR*, abs/2602.10093, 2026. doi: 10.48550/ARXIV.2602.10093. URL https://doi.org/10.48550/arXiv.2602.10093.

Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/2aee1c4159e48407d68fe16ae8e6e49e-Abstract-Conference.html.

Zhuoqun Chen, Xiu Yuan, Tongzhou Mu, and Hao Su. Responsive noise-relaying diffusion policy: Responsive and efficient visuomotor control. *Trans. Mach. Learn. Res.*, 2025, 2025. URL https://openreview.net/forum?id=LLWJkR6gaI.

Zhengxue Cheng, Yiqian Zhang, Anni Tang, Keyu Wang, Wenkang Zhang, Haoyu Li, Hengdi Zhang, and Li Song. Omnivtla: Vision-tactile-language-action models with semantic-aligned tactile sensing. *IEEE Robotics Autom. Lett.*, 11(9):10146–10153, 2026. doi: 10.1109/LRA.2026.3710060. URL https://doi.org/10.1109/LRA.2026.3710060.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *Int. J. Robotics Res.*, 44(10-11):1684–1704, 2025. doi: 10.1177/02783649241273668. URL https://doi.org/10.1177/02783649241273668.

Ruoxuan Feng, Jiangyu Hu, Wenke Xia, Tianci Gao, Ao Shen, Yuhao Sun, Bin Fang, and Di Hu. Anytouch: Learning unified static-dynamic representation across multiple visuo-tactile sensors. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=XToAemis1h.

Carolina Higuera, Akash Sharma, Chaithanya Krishna Bodduluri, Taosha Fan, Patrick Lancaster, Mrinal Kalakrishnan, Michael Kaess, Byron Boots, Mike Lambeta, Tingfan Wu, and Mustafa Mukadam. Sparsh: Self-supervised touch representations for vision-based tactile sensing. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*, volume 270 of *Proceedings of Machine Learning Research*, pp. 885–915. PMLR, 2024. URL https://proceedings.mlr.press/v270/higuera25a.html.

Sigmund H. Høeg, Yilun Du, and Olav Egeland. Fast policy synthesis with variable noise diffusion models. In *IEEE International Conference on Robotics and Automation, ICRA 2025, Atlanta, GA, USA, May 19-23, 2025*, pp. 4821–4828. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11127858. URL https://doi.org/10.1109/ICRA55743.2025.11127858.

Binghao Huang, Yixuan Wang, Xinyi Yang, Yiyue Luo, and Yunzhu Li. 3d-vitac: Learning fine-grained manipulation with visuo-tactile sensing. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2557–2578. PMLR, 2024. URL https://proceedings.mlr.press/v270/huang25e.html.

Jialei Huang, Shuo Wang, Fanqi Lin, Yihang Hu, Chuan Wen, and Yang Gao. Tactile-vla: Unlocking vision-language-action model's physical knowledge for tactile generalization. *CoRR*, abs/2507.09160, 2025. doi: 10.48550/ARXIV.2507.09160. URL https://doi.org/10.48550/arXiv.2507.09160.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *CoRR*, abs/2504.16054, 2025. doi: 10.48550/ARXIV.2504.16054. URL https://doi.org/10.48550/arXiv.2504.16054.

Xiaoqi Li, Muhe Cai, Jiadong Xu, Juan Zhu, Hongwei Fan, Yan Shen, Guangrui Ren, and Hao Dong. AT-VLA: adaptive tactile injection for enhanced feedback reaction in vision-language-action models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2026, Denver, CO, USA, June 3-7, 2026*, pp. 28764–28774. Computer Vision Foundation, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Li_AT-VLA_Adaptive_Tactile_Injection_for_Enhanced_Feedback_Reaction_in_Vision-Language-Action_CVPR_2026_paper.html.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *CoRR*, abs/2210.02747, 2022. doi: 10.48550/ARXIV.2210.02747. URL https://doi.org/10.48550/arXiv.2210.02747.

Yuxiang Lu, Zhe Liu, Xianzhe Fan, Zhenya Yang, Jinghua Hou, Junyi Li, Kaixin Ding, and Hengshuang Zhao. FASTER: rethinking real-time flow vlas. *CoRR*, abs/2603.19199, 2026. doi: 10.48550/ARXIV.2603.19199. URL https://doi.org/10.48550/arXiv.2603.19199.

Dantong Niu, Zhuoyang Liu, Zekai Wang, Boning Shao, Zhao-Heng Yin, Anirudh Pai, Yuvan Sharma, Stefano Saravalle, Ruijie Zheng, Jing Wang, Ryan Punamiya, Mengda Xu, Yuqi Xie, Yunfan Jiang, Letian Fu, Konstantinos Kallidromitis, Matteo Gioia, Junyi Zhang, Jiaxin Ge, Haiwen Feng, Fabio Galasso, Wei Zhan, David M. Chan, Yutong Bai, Roei Herzig, Jiahui Lei, Li Fei-Fei, Ken Goldberg, Jitendra Malik, Pieter Abbeel, Yuke Zhu, Danfei Xu, Linxi Fan, and Trevor

Darrell. T-rex: Tactile-reactive dexterous manipulation. *CoRR*, abs/2606.17055, 2026. doi: 10.48550/ARXIV.2606.17055. URL https://doi.org/10.48550/arXiv.2606.17055.

David Ruhe, Jonathan Heek, Tim Salimans, and Emiel Hoogeboom. Rolling diffusion models. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42818–42835. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/ruhe24a.html.

Tong Wu, Zhihao Fan, Xiao Liu, Hai-Tao Zheng, Yeyun Gong, Yelong Shen, Jian Jiao, Juntao Li, Zhongyu Wei, Jian Guo, Nan Duan, and Weizhu Chen. Ar-diffusion: Auto-regressive diffusion model for text generation. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/7d866abba506e5a56335e4644ebe18f9-Abstract-Conference.html.

Han Xue, Jieji Ren, Wendi Chen, Gu Zhang, Yuan Fang, Guoying Gu, Huazhe Xu, and Cewu Lu. Reactive diffusion policy: Slow-fast visual-tactile policy learning for contact-rich manipulation. *CoRR*, abs/2503.02881, 2025. doi: 10.48550/ARXIV.2503.02881. URL https://doi.org/10.48550/arXiv.2503.02881.

Chengbo Yuan, Zicheng Zhang, Mingjie Zhou, Wendi Chen, Yi Wang, Zhuoyang Liu, Dantong Niu, Shuo Wang, Hui Zhang, Wenkang Zhang, Yingdong Hu, Yuanqing Gong, Wanli Xing, Chuan Wen, Cewu Lu, Kaifeng Zhang, and Yang Gao. FTP-1: A generalist foundation tactile policy across tactile sensors for contact-rich manipulation. *CoRR*, abs/2606.13102, 2026. doi: 10.48550/ARXIV.2606.13102. URL https://doi.org/10.48550/arXiv.2606.13102.

Yujie Zang, Yuhang Zheng, Xian Nie, Yupeng Zheng, Shuai Tian, Songen Gu, Chen Gao, Zining Wang, Shuicheng Yan, and Wenchao Ding. Tacforesight: Force-guided tactile world model for contact-rich manipulation. *CoRR*, abs/2606.11184, 2026. doi: 10.48550/ARXIV.2606.11184. URL https://doi.org/10.48550/arXiv.2606.11184.

Shiqi Zhang, Xin Zhang, Yedong Shen, Yao Li, Yuxuan Gao, Sha Zhang, Yuan Zhang, Kaixue Long, Jiajia Wu, Jia Pan, Jiajun Deng, and Yanyong Zhang. Retouch: Empowering contact-rich dexterous manipulation with online-refined tactile prediction. *CoRR*, abs/2608.01824, 2026. doi: 10.48550/ARXIV.2608.01824. URL https://doi.org/10.48550/arXiv.2608.01824.

Zihan Zhang, Richard Liu, Rana Hanocka, and Kfir Aberman. Tedi: Temporally-entangled diffusion for long-term motion synthesis. In Andres Burbano, Denis Zorin, and Wojciech Jarosz (eds.), *ACM SIGGRAPH 2024 Conference Papers, SIGGRAPH 2024, Denver, CO, USA, 27 July 2024-1 August 2024*. ACM, 2024. doi: 10.1145/3641519.3657515. URL https://doi.org/10.1145/3641519.3657515.

Jialiang Zhao, Yuxiang Ma, Lirui Wang, and Edward H. Adelson. Transferable tactile transformers for representation learning across diverse sensors and tasks. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3766–3779. PMLR, 2024. URL https://proceedings.mlr.press/v270/zhao25c.html.

Jianyi Zhou, Feiyang Hong, Yunhao Li, Yicheng Zhao, Yongjue Cen, Zirui Liu, Jiakang Huang, Zirui Chen, Ruiyang Zhang, Weizhuo Zhu, Xuhua Song, and Shuo Yang. Touchworld: A predictive and reactive tactile foundation model for dexterous manipulation. *CoRR*, abs/2607.07287, 2026. doi: 10.48550/ARXIV.2607.07287. URL https://doi.org/10.48550/arXiv.2607.07287.

A Additional Implementation Details

A.1 Simulation Training Parameters

We train on 50 demonstration trajectories per task for 15,000 steps with a global batch size of 256. We use AdamW with a peak learning rate of 5×10^{-5}, a cosine decay to 5×10^{-6}, 2,000 warm-up steps, a weight decay of 10^{-10}, and gradient clipping at a global norm of 1.0. We use $H = 50$, $K = 10$, and $B = 5$, with $S = 5$ at inference ($N = KS = 50$).

A.2 Real-World Training Parameters

We use 100 demonstration trajectories per task, except for *Close Cap* and *Close Lid*, which use 150 and 300 trajectories, respectively. We train for 15 epochs with a global batch size of 256. We use AdamW with a peak learning rate of 6×10^{-5}, a cosine decay to 1×10^{-6}, 2,000 warm-up steps, a weight decay of 10^{-5}, and gradient clipping at a global norm of 1.0. We use $H = 40$, $K = 8$, and $B = 5$, with $S = 2$ at inference ($N = KS = 16$).

A.3 Representation-Dynamics Analysis

We analyze a representative dropper-squeezing episode over a 40-action horizon, sampled every five actions at 30 FPS. Visual representations are extracted separately for the top and wrist cameras by averaging the final spatial tokens from the frozen visual encoder of GR00T N1.7 (Bjorck et al., 2025). Tactile representations are extracted separately for the thumb and index finger from the final output of the tactile encoder in the trained real-world TacForcing model. This encoder is initialized from the pretrained tactile encoder of FTP-1 (Yuan et al., 2026). Both encoders use their native evaluation preprocessing, with no additional feature normalization before computing cosine distance.

For sensor j and sampling offset $k \in \{0, 5, \ldots, 35\}$, we compute the cosine distance between the current representation $\mathbf{z}_k^{(j)}$ and its initial representation $\mathbf{z}_0^{(j)}$. The modality-level distance is obtained by averaging the resulting distances across the two corresponding sensors:

$$d_k^{(j)} = 1 - \frac{(\mathbf{z}_0^{(j)})^\top \mathbf{z}_k^{(j)}}{\|\mathbf{z}_0^{(j)}\|_2 \|\mathbf{z}_k^{(j)}\|_2}, \qquad D_k^{(q)} = \frac{1}{|\mathcal{S}_q|} \sum_{j \in \mathcal{S}_q} d_k^{(j)}, \tag{9}$$

where $\mathcal{S}_{\text{vis}}$ contains the top and wrist cameras, and $\mathcal{S}_{\text{tac}}$ contains the thumb and index finger. The values 0.005 and 0.55 reported in the main text are $D_{35}^{(\text{vis})}$ and $D_{35}^{(\text{tac})}$, respectively, at the final sampled offset rather than averages over time.

B Real-World Platform and Task Settings

B.1 Real-World Platform

We evaluate on Sharpa North and Sharpa & RealMan (Figure 4). The Sharpa & RealMan platform consists of two 7-DoF RealMan RM75 robot arms, each equipped with a 22-DoF Sharpa Wave dexterous hand. Tactile sensors embedded in the fingertips provide deformation maps during contact. A top-mounted RealSense camera captures the global workspace, while wrist-mounted RealSense cameras provide local views of object interactions. Manus Pro data gloves are used in the demonstration-collection interface. Figure 6 shows the complete setup and its main components.

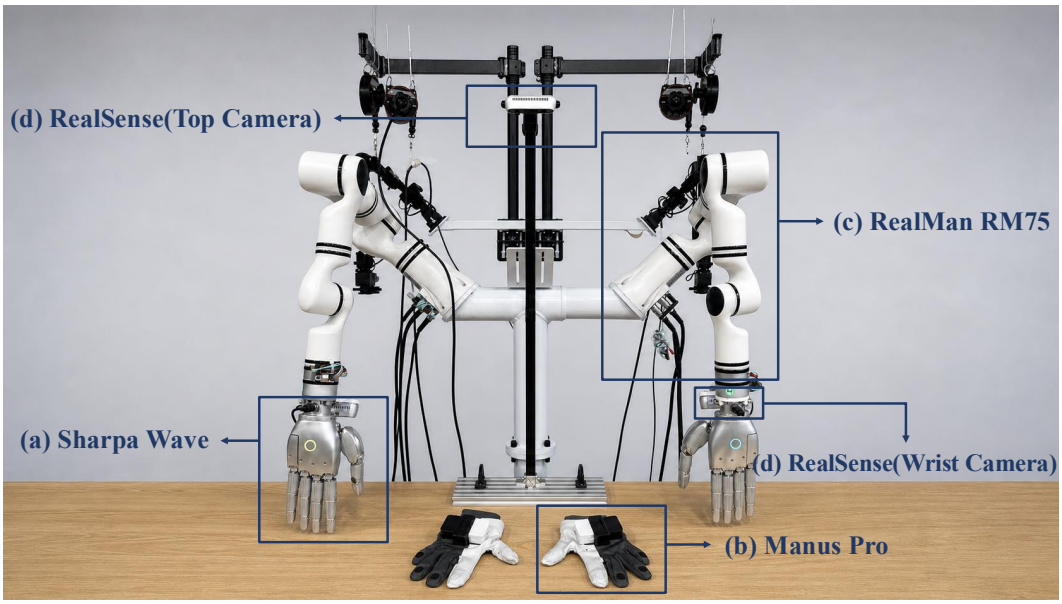


Figure 6: **Real-world platform.** The setup comprises (a) Sharpa Wave dexterous hands, (b) Manus Pro data gloves, (c) two RealMan RM75 robot arms, and (d) top- and wrist-mounted RealSense cameras.

B.2 Real-World Task Settings

We evaluate six contact-rich manipulation tasks across the two platforms shown in Figure 4. *Close Cap*, *Insert Battery*, and *Close Lid* are performed on Sharpa North; *Transfer Liquid*, *Wipe Board*, and *Stand Bottle* are performed on Sharpa & RealMan. Figures 7 and 8 provide execution sequences for the two platforms in the order listed in Table 2. Each method is evaluated over 16 independent trials per task.

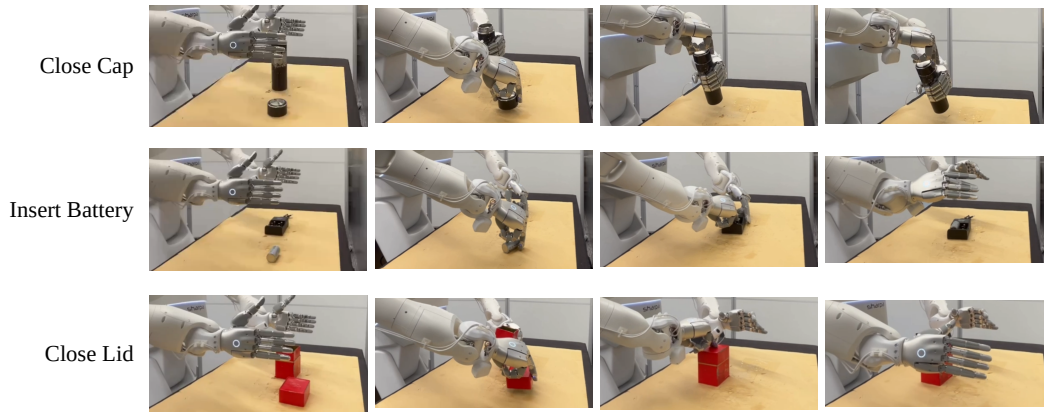


Figure 7: **Sharpa North tasks.** Representative execution sequences in the task order of Table 2. Each row progresses from left to right.

Close Cap requires rotating a cap onto a drinking cup; *Insert Battery* requires aligning and seating a battery in its compartment; *Close Lid* requires aligning and pressing the lid onto a tea box.

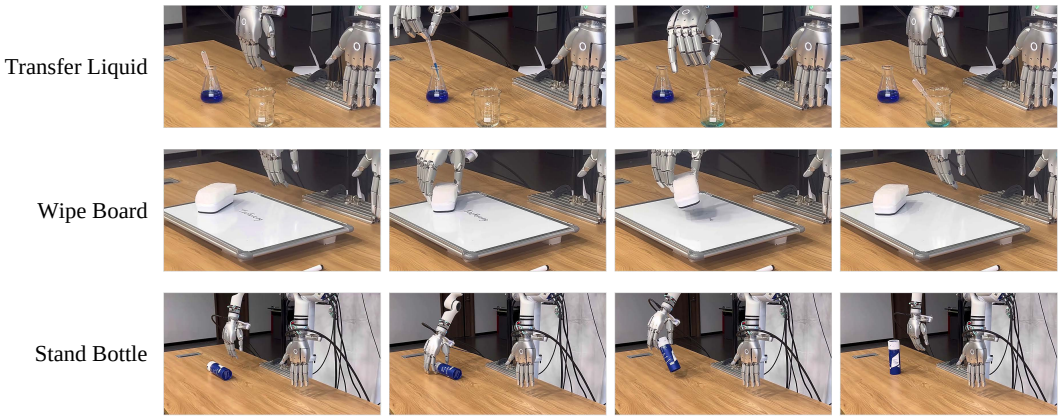


Figure 8: **Sharpa & RealMan tasks.** Representative execution sequences in the task order of Table 2. Each row progresses from left to right.

Transfer Liquid uses a dropper to move liquid between containers; *Wipe Board* requires sustained contact with a whiteboard; *Stand Bottle* requires reorienting a horizontal bottle and placing it upright.

C Simulation Task Details

We evaluate TacForcing on six tasks from the UniVTAC benchmark (Chen et al., 2026). Figure 9 shows representative execution sequences for these tasks.

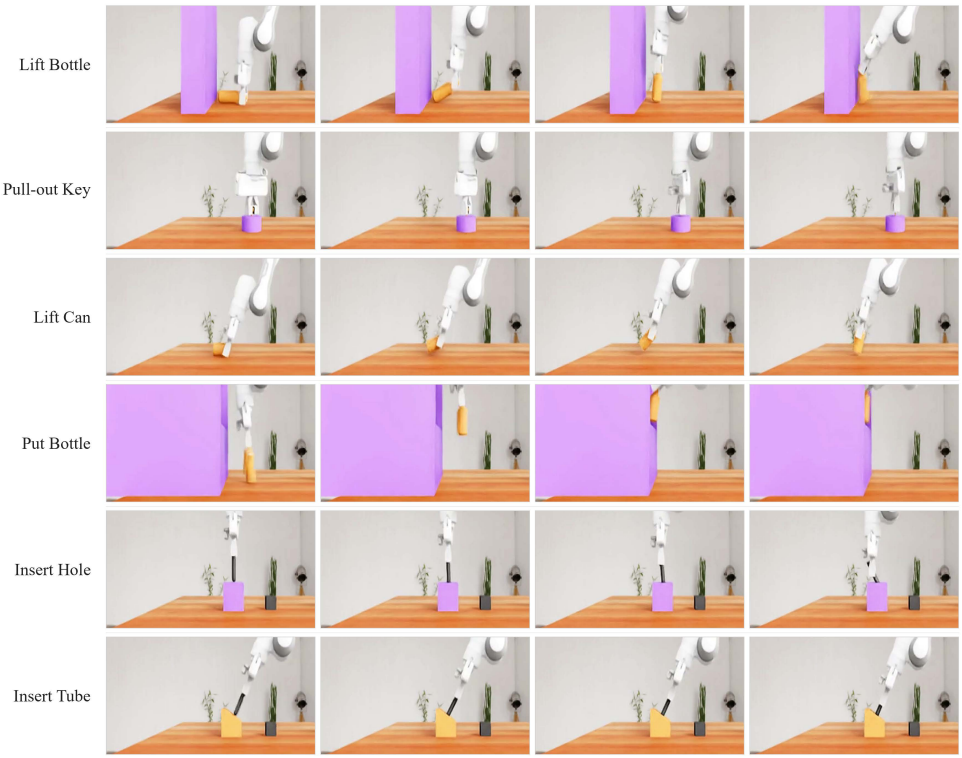


Figure 9: **UniVTAC simulation tasks.** Representative execution sequences for the six tasks used in our simulation experiments, reproduced from UniVTAC (Chen et al., 2026).