TF^2-STEERING: ACTION REFINEMENT WITH TACTILE FORECASTS AND FEEDBACK VIA VELOCITY-FIELD STEERING

Anonymous authors
Paper under double-blind review

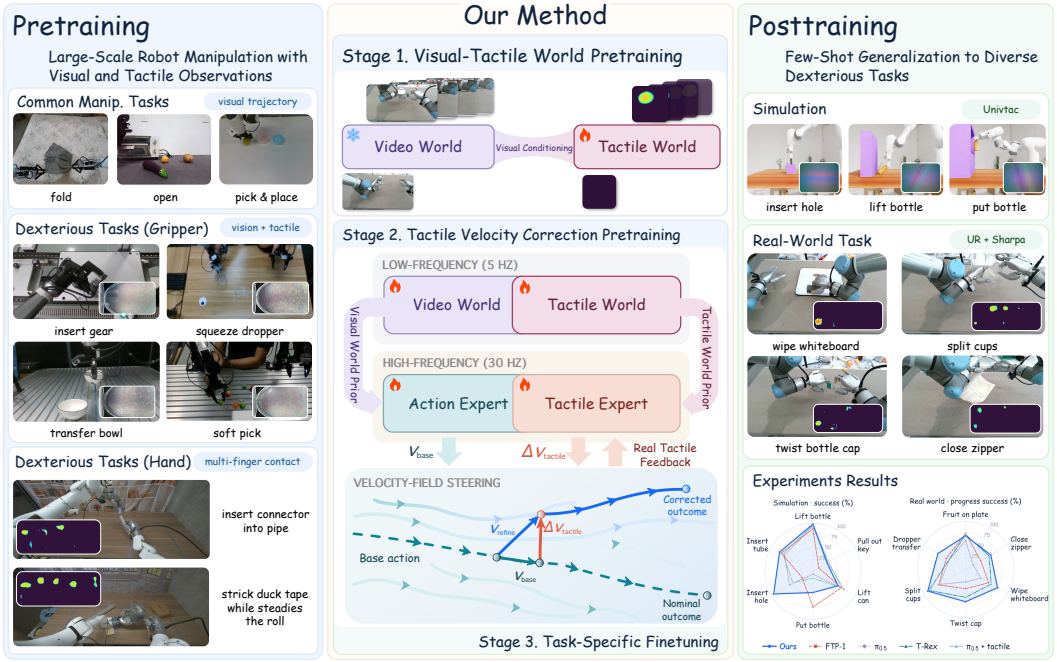

Figure 1: Overview of TF^2-Steering. **Left:** visual-only and visual–tactile pretraining data. **Middle:** the tactile expert uses cached forecasts from the tactile world model and real-time tactile feedback to predict delta velocity, refining the action expert's flow-matching velocity. The video world model and action expert remain independently usable without tactile input. **Right:** task-specific post-training and evaluation in simulation and the real world.

ABSTRACT

Tactile sensing plays a crucial role in everyday human interaction. Consider inserting a plug into a tight socket, vision alone provides limited guidance during the final insertion stage, and we instead rely on both anticipated contact and realized tactile feedback to ajust the insertion motion. The same capability is equally important for robots, particularly in dexterous manipulation tasks. Yet many contemporary robot policies remain vision-dominant. This visual bias, compounded by limited tactile data and sparse contact events, can cause directly incorporated tactile signals to be ignored or even compromise the policy's pretrained capabilities, yielding degrading performance as observed in our experiments. We present TF^2-Steering, which refines actions via learned residual steering of the flow-matching action expert's velocity field, using a tactile mixture-of-experts architecture to integrate both tactile forecasts and real-time tactile feedback. Together with specific designed three stages training recipe, TF^2-Steering enables tactile signals to correct and, when necessary, redirect the flow-matching action-generation process, while building on the strong action-generation capacity of the vision-based foundation

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.

policy. Extensive experiments across six simulated and six real-world manipulation tasks demonstrate the effectiveness of TF^2-Steering. It outperforms the strongest evaluated baselines by 10.7 percentage points in average simulation success rate and 9.1 percentage points in average real-world progress success rate.

1 INTRODUCTION

Recent robot policy models, particularly vision-language-action (VLA) and world-action (WA) foundation policies (Physical Intelligence et al., 2025; Ma et al., 2026a; Cai et al., 2026; Zhang et al., 2026b), have advanced rapidly and demonstrated strong generalization across tasks, objects, and environments. However, their capabilities remain limited in contact-rich manipulation, where task success depends on transient physical interactions that are difficult to infer reliably from visual observations alone. Tactile sensing provides a complementary source of information about these interactions, making it especially valuable for dexterous manipulation. Accordingly, recent studies have begun incorporating tactile signals into robot foundation policies and have reported promising results on dexterous and precision contact-rich tasks (Yu et al., 2026; Lou et al., 2026; Li et al., 2026b; Fu et al., 2026; Cheng et al., 2026; Yuan et al., 2026a). These results highlight tactile sensing as an important direction for extending the capabilities of general-purpose robot policies.

Yet two important questions have not been sufficiently discussed. First, **how can tactile information be incorporated into a vision-dominant foundation policy?** Robot foundation models are trained on large-scale demonstration data and acquire strong visual action-planning abilities. The temporal sparsity of tactile signals and the limited scale of tactile data relative to visual data may cause a vision-dominant policy to ignore tactile cues when they are naively appended during fine-tuning, or disturb the pretrained visual representation and reduce generalization to settings without tactile sensors (Liu et al., 2025; Chen et al., 2026b; Zhang et al., 2026d). Recent approaches therefore isolate tactile influence through residual pathways or gating mechanisms (Zhou et al., 2026b; Li et al., 2026a). However, Residual pathways typically introduce a separately trained correction branch that can only modify an already generated action, limiting its ability to redirect a suboptimal plan. Gating mechanisms preserve the backbone, but their learned gates may attenuate tactile features before they reach the action head, making the tactile contribution weak or inconsistent. Second, **should tactile sensing be treated solely as feedback, or also modeled predictively?** Human manipulation relies on both: prior experience allows us to anticipate an appropriate grasp configuration and force from visual context, while realized contact feedback enables online adjustment during execution. Existing studies have explored reactive tactile feedback (He et al., 2025; Xue et al., 2025; Niu et al., 2026) and future tactile prediction (Heng et al., 2025; Yuan et al., 2026b; Lou et al., 2026), but these roles have not yet been consistently integrated into a single causal action-correction process. We posit that reliable dexterous manipulation depends on both the expected and realized contact for action refinement.

In this work, we introduce TF^2-Steering, a framework that further discusses these two questions. Inspired by recent studies that modify flow-matching velocity fields for controlled generation and policy adaptation (Zhao et al., 2024; Park et al., 2024; Zhang et al., 2025; Patel et al., 2025), we incorporate tactile information by steering the velocity field of the flow-matching action expert. Specifically, TF^2-Steering augments the original action expert with a tactile expert that attends unidirectionally to its features. Using tactile forecasts and real-time tactile feedback, the tactile expert predicts a delta velocity to refine the base flow-matching velocity during action generation, while the original action expert remains independent of tactile input. To complement real-time feedback with tactile anticipation, we introduce a tactile world model that predicts future tactile states from the available world context. TF^2-Steering adopts a dual-rate design: the video and tactile world models provide low-frequency planning priors, while the action and tactile experts refine actions at a higher frequency during execution. Tactile forecasts are generated before each action block and held fixed throughout its execution, allowing the tactile expert to steer action generation using incoming real-time feedback alongside these cached forecasts.

Our main contributions are threefold:

- We develop a dual-rate framework that combines low-frequency tactile forecasts, cached before execution, with high-frequency tactile feedback for online action refinement. A three-

stage training strategy supports both tactile forecasting and the effective use of predicted and real-time tactile information.

- We introduce a tactile-guided velocity-field steering algorithm that learns delta-velocity corrections within the flow-matching action-generation process. This design enables local action refinement while retaining the original action expert as an independently usable source of base actions.

- We evaluate our framework on six simulated and six real-world manipulation tasks, outperforming the strongest evaluated baselines by 10.7 percentage points in average simulation success rate and 9.1 percentage points in average real-world progress success rate. Ablation studies further support the contributions of tactile forecasting, real-time feedback, and velocity-field steering.

2 Related Work

Velocity-Field Adaptation in Flow Matching. Flow Matching models generate samples by integrating a learned time-dependent velocity field. Adapting this field provides a modular way to build upon a pretrained flow model while introducing task-specific control. Existing studies have explored velocity-field adaptation for several purposes. FlowTurbo (Zhao et al., 2024) and Constant Acceleration Flow (Park et al., 2024) introduce velocity residuals or acceleration terms to improve trajectory expressiveness and sampling efficiency, while FlowPolicy (Zhang et al., 2025) and SCoT (Fan et al., 2026) enforce velocity consistency for fast generation. Other works, including FlowChef (Patel et al., 2025), FlowEdit (Kulikov et al., 2025), and DNAEdit (Xie et al., 2026), steer or refine velocity fields to incorporate measurements or perform controlled image editing. Graph Flow Matching (Siddiqui et al., 2026) and Energy Matching (Balcerak et al., 2026) further augment flow dynamics with neighborhood or energy-based priors. Our work extends this paradigm to tactile-conditioned action generation by learning an action-space correction field from contact observations.

Visual-tactile Manipulation. Tactile sensing complements vision by providing localized cues about contact geometry, force, friction, and slip, which are particularly valuable for dexterous manipulation when visual observations are severely occluded. Early studies incorporated tactile or force feedback into task-specific visuomotor policies, including FoAR (He et al., 2025) and Reactive Diffusion Policy (Xue et al., 2025). More recent work has integrated tactile sensing into general-purpose VLA and WA models. One line of research focuses on tactile-language grounding and semantic alignment, linking tactile observations to task-relevant physical concepts (Hao et al., 2025; Zhang et al., 2026a; Huang et al., 2025; Cheng et al., 2026). Another investigates force-aware action generation and tactile-reactive policy architectures through force-conditioned expert routing (Yu et al., 2026), hybrid force-position control (Li et al., 2026b), and slow-fast inference (Niu et al., 2026). Large-scale tactile policy pretraining has also been explored to facilitate the transfer of manipulation skills across heterogeneous sensors and embodiments (Yuan et al., 2026a). However, in vision-dominant policy models, naively appending tactile inputs can leave tactile cues underutilized and degrade the pretrained policy's ability to plan and generate actions, consistent with prior findings on force underutilization and modality imbalance (Liu et al., 2025; Chen et al., 2026b; Zhang et al., 2026d). Existing methods address these challenges through contact-aware gating (Zhang et al., 2026c), adaptive tactile injection (Li et al., 2026a), asymmetric attention (Wu et al., 2026; Tian et al., 2026), and bounded residual action modulation (Wang et al., 2026). Our method applies tactile-driven corrections directly to the flow-matching velocity field, preserving the pretrained policy's visual planning while enabling precise contact-level adjustments.

A complementary line of work predicts future tactile states to learn internal contact grounding, treating tactile prediction either as an auxiliary objective or as a component of a visuotactile world model (Heng et al., 2025; Huang et al., 2026; Yuan et al., 2026b; Zheng et al., 2026; Lou et al., 2026). TacForcing and TacPAC (Zhou et al., 2026a; Ma et al., 2026b) further mitigate the temporal mismatch between chunked action generation and tactile feedback available during execution. Our method brings these two functions together through a tactile expert that jointly uses tactile forecasts and real-time tactile feedback to refine the action expert's flow-matching velocity during action generation.

3 METHOD

Given a task instruction $\mathcal{I}$, the current multi-view visual observation $\mathbf{o}_t^{\mathrm{v}}$, and the current tactile observation $\mathbf{o}_t^{\tau}$, TF^2-Steering produces an action chunk $\mathbf{A}_{t:t+H-1}$, where t denotes the current control step, H is the planning horizon, and the superscripts v and τ indicate the visual and tactile modalities, respectively. During closed-loop execution, real-time tactile feedback is passed to the tactile expert to refine the flow-matching velocity for the next unexecuted action block. In the remainder of this section, we describe the TF^2-Steering model architecture, the asynchronous velocity-field steering algorithm, and the three-stage training procedure.

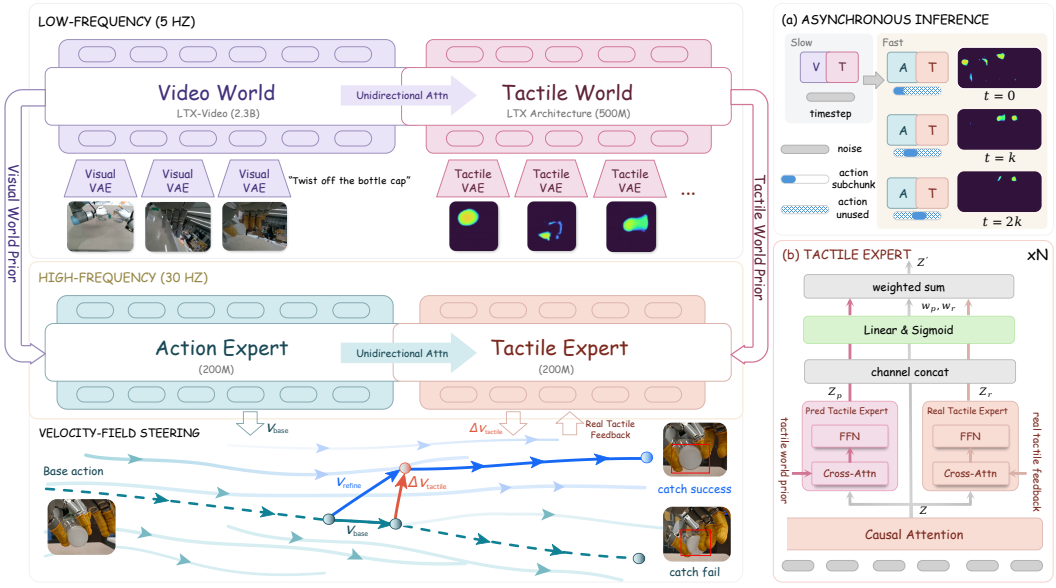

Figure 2: **Overview of TF^2-Steering.** The video and tactile world models provide low-frequency planning priors, while the action and tactile experts perform high-frequency action refinement through learned delta-velocity corrections. Unidirectional attention keeps the original world-action backbone independent of tactile input. **(a)** Asynchronous inference uses incoming tactile feedback to refine subsequent action blocks, with pre-execution tactile forecasts held fixed throughout each block. **(b)** The tactile expert integrates predicted and real-time tactile features through separate subexperts with sigmoid weighting to steer the action expert's flow-matching velocity field.

3.1 MODEL ARCHITECTURE

As shown in Fig. 2, TF^2-Steering consists of a pretrained world-action model, a tactile world model, and a forecast–feedback tactile correction expert. The world-action model provides the visual context and base action generation, while the tactile world model and tactile correction expert enable tactile-guided action refinement.

World-Action Model Backbone. The world-action model comprises LTX-Video (HaCohen et al., 2024), a modality compression Q-Former, and a flow-matching action expert. To better accommodate the robot's multi-view observations, we augment LTX-Video with a multi-view cross-attention module. We train the video model with a single-step denoising objective by setting the flow-matching noise ratio to 1 during training. This design allows the video model to perform inference with a single forward pass. The Q-Former subsequently compresses the visual context into compact query features, which condition the action expert for action generation. As a result, the model generates an action chunk in 240 ms on a single NVIDIA RTX 4090. Additional training and implementation details are provided in Appendix A.1.1.

Tactile World Model. The tactile world model is built on the LTX-Video architecture, but uses a smaller model with approximately 500M parameters. We remove the multi-view cross-attention module, which is unnecessary for tactile images, and reduce the model width and depth while retaining

roughly one-third of the original layers. The tactile world model is connected to LTX-Video through a unidirectional attention pathway: the tactile predictor can attend to visual features, whereas the visual stream cannot attend to tactile features. During training, the model receives the current tactile image and predicts a future tactile observation using the same single-step denoising formulation as the video model. To provide a suitable latent representation for tactile inputs, we first fine-tune the LTX-Video VAE on tactile videos and then use the resulting VAE to encode tactile images. At inference time, rather than decoding the predicted tactile image, we use the feature from the final layer of the tactile world model as the tactile-world prior for the downstream tactile expert. This feature-level interface reduces inference cost while retaining richer task-related context than a decoded tactile image.

Forecast-Feedback Tactile Correction Expert. We introduce a forecast–feedback tactile correction expert, referred to as the *tactile expert* hereafter, to integrate both predicted tactile features from the tactile world model and real-time tactile features. The tactile expert attends unidirectionally to the action expert's features, with stop-gradient applied before these features enter the tactile expert. This allows the tactile expert to access action context while keeping tactile inputs outside the action expert. As detailed in Fig. 2(b), each tactile expert block contains separate predicted-tactile and real-tactile subexperts, inspired by mixture-of-experts architectures (Fedus et al., 2022; Jiang et al., 2024). We use sigmoid weighting to modulate the two branches independently, without constraining their weights to sum to one.

The tactile expert and the action expert receive the same noised action $\widetilde{\mathbf{a}}_s$ and flow-matching timestep s. Let $\mathbf{v}_\theta^{\text{base}}$ denote the base flow-matching velocity produced by the action expert, $\Delta\mathbf{v}_\phi$ denote the correction predicted by the tactile expert, and $\mathbf{v}^\star$ denote the flow matching target. As illustrated in the lower-left portion of Fig. 2, the tactile expert steers action generation by refining the base velocity:

$$\mathbf{v}^{\text{corr}} = \mathbf{v}_\theta^{\text{base}} + \Delta\mathbf{v}_\phi. \tag{1}$$

The action and tactile experts are trained with separately computed flow matching losses:

$$\mathcal{L}_{\text{act}} = \left\|\mathbf{v}_\theta^{\text{base}} - \mathbf{v}^\star\right\|_2^2, \qquad \mathcal{L}_{\text{steer}} = \left\|\mathbf{v}^\star - \left(\text{sg}\left[\mathbf{v}_\theta^{\text{base}}\right] + \Delta\mathbf{v}_\phi\right)\right\|_2^2, \tag{2}$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operator. Together with the stop-gradient on the base velocity in the steering objective, detaching the action features prevents the steering loss from updating the action expert, which remains trainable through its own action loss.

3.2 Asynchronous Velocity-Field Steering

To support high-frequency tactile feedback and action correction, TF^2-Steering uses a two-rate control system. The video world model and tactile world model form a low-frequency planning system that generates long-horizon visual and tactile predictions, whereas the action expert and tactile expert form a high-frequency steering system that updates the action sequence during execution.

Figure 2(a) illustrates the asynchronous generation and execution of an action chunk containing H control steps, executed in blocks of k steps. The low-frequency system performs inference once to generate the chunk, and the resulting visual and tactile world priors remain fixed throughout its execution and all subsequent high-frequency updates. After each subchunk k is executed, the high-frequency system receives the latest tactile feedback and regenerates the remaining actions. During every flow-matching iteration of this regeneration process, the tactile expert continues to correct the action expert's base velocity. Actions that have already been executed are discarded, and the robot executes only the newly generated actions that remain unexecuted.

During training, we randomly sample a delay $0 \le d < H$ to simulate the delayed arrival of tactile feedback in the high-frequency system. At time t, the video world model receives $\mathbf{o}_t^{\text{v}}$ and the tactile world model receives $\mathbf{o}_t^{\tau}$, while the real-time tactile observation $\mathbf{o}_{t+d}^{\tau,\text{real}}$ is provided to the tactile expert after the sampled delay. The tactile correction loss is evaluated only within a steering window after the delay. Specifically, for

$$\mathcal{W}_d = \{i \,|\, d \le i < \min(d+W, H)\}, \tag{3}$$

we define

$$\mathcal{L}_{\text{steer}} = \frac{1}{|\mathcal{W}_d|} \sum_{i \in \mathcal{W}_d} \left\|\mathbf{v}_i^\star - \left(\text{sg}\left[\mathbf{v}_{\theta,i}^{\text{base}}\right] + \Delta\mathbf{v}_{\phi,i}\right)\right\|_2^2. \tag{4}$$

We provide the tactile expert with a relative-position encoding $\rho_i = i - d$ to locate each action relative to the feedback time. Together with the windowed loss, this encoding associates tactile feedback with the corresponding action window. The block length k can be adjusted at inference time to control the refinement frequency.

3.3 Training

TF^2-Steering uses large-scale tactile pretraining for both the tactile world model and the tactile expert, providing a stronger initialization for generalization across diverse contact conditions.

Tactile Prediction Pretraining. In the first stage, we pretrain the tactile world model on large-scale tactile datasets, including OpenNeoData (Team et al., 2026), the T-REX dataset (Niu et al., 2026), and FTP-1-Sharpa (Yuan et al., 2026a). The video world model provides the visual conditioning feature $\mathbf{h}_t^{\mathrm{v}}$ but is kept fixed throughout this stage. Given the current tactile observation $\mathbf{o}_t^{\tau}$, a noised VAE latent $\widetilde{\mathbf{x}}_s^{\tau}$ of a future tactile observation, and the flow-matching timestep s, the tactile world model is trained with

$$\mathcal{L}_{\mathrm{pred}}^{\tau} = \mathbb{E}\left[\left\|\mathbf{v}^{\tau,\star} - \mathbf{v}_{\psi}^{\tau}\left(\widetilde{\mathbf{x}}_s^{\tau}, s \mid \mathbf{o}_t^{\tau}, \mathbf{h}_t^{\mathrm{v}}\right)\right\|_2^2\right], \tag{5}$$

where $\mathbf{v}^{\tau,\star}$ denotes the flow-matching target in the VAE latent space of the future tactile observation.

Tactile Velocity Steering Pretraining. In the second stage, we train all modules end-to-end on the same tactile data to enable the tactile expert to use both predicted tactile features and real-time tactile feedback for action correction. To preserve the generalization of the world-action backbone, we additionally mix in 10% tactile-free samples from the original world-action pretraining data. Let $\mathcal{L}_{\mathrm{pred}}^{\mathrm{v}}$ denote the video-world prediction loss. The joint training objective is

$$\mathcal{L}_{\mathrm{joint}} = \mathcal{L}_{\mathrm{pred}}^{\mathrm{v}} + \mathcal{L}_{\mathrm{pred}}^{\tau} + \mathcal{L}_{\mathrm{act}} + \lambda_{\mathrm{steer}}\mathcal{L}_{\mathrm{steer}}. \tag{6}$$

Here, λ_{steer} controls the contribution of the steering loss. Further details of the data mixture and training configuration are provided in Appendix A.1.2.

Task-Specific Fine-Tuning. In the final stage, we retain the same training objective as in the tactile velocity steering pretraining stage and perform task-specific post-training on both simulated and real-robot data. This procedure adapts the pretrained model to the target manipulation tasks and yields clear performance gains over training without tactile pretraining.

4 Experiments

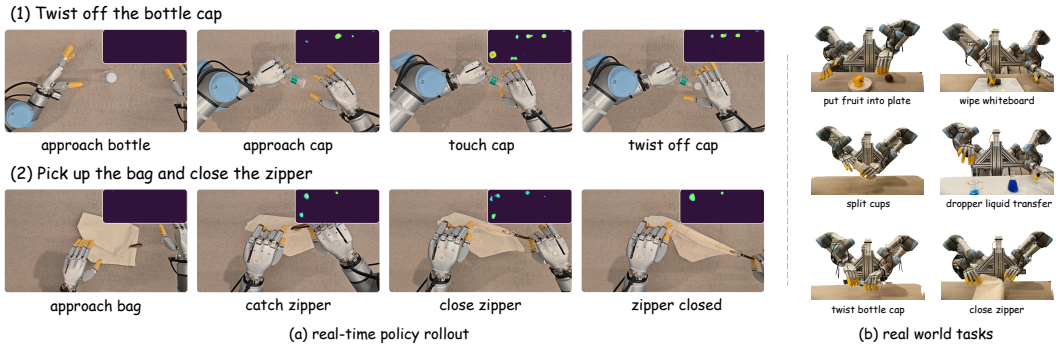

Figure 3: Real-world manipulation tasks and example execution sequences. **(a)** Bottle-cap twisting and zipper closing, with tactile visualizations shown as insets. **(b)** Task scenes for fruit placement, whiteboard wiping, cup separation, liquid transfer with a dropper, bottle-cap twisting, and zipper closing.

4.1 Experiment Setup

Tasks and Evaluation Metrics. We evaluate TF^2-Steering in both simulation and real-world experiments. For simulation, we use UniVTAC (Chen et al., 2026a), a benchmark for contact-rich

Table 1: **Simulation results on UniVTAC.** Success rate (%, ↑) on six tasks; Avg. denotes the macro-average. **Bold** and underlined values indicate the best and second-best results in each column, respectively, including ties. *ACT and VITaL results are taken from UniVTAC (Chen et al., 2026a).

Method	Lift Bottle	Pull Out Key	Lift Can	Put Bottle	Insert Hole	Insert Tube	Avg.
ACT* (Zhao et al., 2023)	71.0	46.0	29.0	31.0	25.0	34.0	39.3
VITaL* (George et al., 2025)	72.0	**47.0**	8.0	32.0	25.0	34.0	36.3
FTP-1 (Yuan et al., 2026a)	88.0	14.0	68.0	**74.0**	26.0	76.0	57.7
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	94.0	36.0	**74.0**	6.0	56.0	68.0	55.7
$\pi_{0.5}$ + tactile	96.0	38.0	68.0	14.0	48.0	72.0	56.0
TF^2-Steering (ours)	**98.0**	34.0	60.0	44.0	**94.0**	**80.0**	**68.3**

Table 2: **Real-world results.** Progress success rate (%, ↑) over 20 trials per task; Avg. denotes the macro-average over the six tasks. **Bold** and underlined values indicate the best and second-best results in each column, respectively.

Method	Fruit on Plate	Close Zipper	Wipe Whiteboard	Twist Cap	Split Cups	Dropper Transfer	Avg.
ViTacFormer (Heng et al., 2025)	12.5	4.0	2.0	9.0	7.0	2.0	6.1
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	60.0	18.0	36.0	26.0	62.0	39.0	40.2
$\pi_{0.5}$ + tactile	58.0	12.0	39.0	22.0	57.0	23.0	35.2
T-Rex (Niu et al., 2026)	72.5	56.0	61.0	58.0	87.0	40.0	62.4
FTP-1 (Yuan et al., 2026a)	**75.0**	36.0	55.0	40.0	64.0	46.0	52.7
TF^2-Steering (ours)	70.0	**60.0**	**76.0**	**67.0**	**90.0**	**66.0**	**71.5**

manipulation that includes tasks such as lifting a bottle and pulling out a key. Since some UniVTAC tasks support only single-camera observations, we train two checkpoints for each method: one for the single-view task group and one for the multi-view task group (Appendix A.2.2). For real-world experiments, we use a bimanual platform comprising two 6-DoF UR5e arms, each equipped with a 22-DoF Sharpa dexterous hand (Appendix A.2.1). We collect 100 demonstrations for each of six tasks: putting fruit on a plate, wiping a whiteboard, splitting cups, transferring liquid with a dropper, twisting a bottle cap, and closing a zipper (Fig. 3b). All baselines undergo task-specific training on the collected demonstrations, with pretrained models adapted through post-training. We evaluate each method over 20 trials per real-world task and report progress success rate to capture partial task completion and provide a more fine-grained assessment of manipulation performance. The scoring rules are detailed in Appendix A.2.4.

Baseline. We compare TF^2-Steering with five baselines: (1) **ViTacFormer** (Heng et al., 2025), an ACT-based policy trained separately for each task that predicts future tactile force and torque signals to support action generation; (2) $\pi_{0.5}$, a vision-language-action (VLA) model pretrained on large-scale, diverse data; (3) $\pi_{0.5}$ **+ tactile**, our implementation that concatenates all tactile images into a single composite image and supplies it as an additional image input to $\pi_{0.5}$, testing the effect of directly incorporating tactile observations on task performance; (4) **T-Rex** (Niu et al., 2026), a tactile-reactive VLA system that incorporates high-frequency tactile feedback during inference through cascaded flow matching for action refinement; and (5) **FTP-1** (Yuan et al., 2026a), a foundation tactile policy pretrained on large-scale tactile data that unifies heterogeneous tactile signals and integrates them into a VLA architecture. Further details on baseline training and adaptation are provided in Appendix A.2.3.

4.2 Main Results

Simulation. As shown in Table 1, TF^2-Steering achieves an average success rate of 68.3%, outperforming the strongest baseline, FTP-1, by 10.6 percentage points. The gains are particularly pronounced on Insert Hole, where our method achieves 94.0% success compared with 56.0% for

Table 3: **Model design ablations.** Real-world progress success rate (%, ↑) on three manipulation tasks; Avg. denotes the macro-average. Δ Avg. reports the change relative to the full model in percentage points (pp), computed before rounding; negative values indicate a decline. **Bold** indicates the best result in each task and average column, including ties.

Configuration	Twist Cap	Close Zipper	Split Cups	Avg.	Δ Avg. (pp)
TF^2-Steering (full model)	**67.0**	**60.0**	**90.0**	**72.3**	–
Tactile modalities					
w/o predicted tactile	61.0	58.0	**90.0**	69.7	−2.7
w/o real-time tactile	54.0	49.0	86.0	63.0	−9.3
w/o both tactile modalities	51.0	50.0	84.0	61.7	−10.7
Tactile integration strategy					
Action-space residual correction	53.0	49.0	82.0	61.3	−11.0
Architecture					
w/o asynchronous inference	56.0	54.0	**90.0**	66.7	−5.7
w/o attention masking & stop-gradient	49.0	51.0	88.0	62.7	−9.7

the strongest baseline, while also attaining the highest success rate on Insert Tube. These results support the effectiveness of tactile-guided action refinement for precise insertion. In comparison, directly incorporating tactile images into $\pi_{0.5}$ yields little improvement in average success rate (56.0% versus 55.7%), suggesting that simply providing tactile observations is insufficient for effective use during action generation. This may reflect the difficulty of translating tactile signals supplied as image observations into appropriate action corrections, given that the model's large-scale pretraining primarily emphasizes vision-driven action planning.

Real World Tasks. Table 2 shows that TF^2-Steering achieves an average progress success rate of 71.5%, exceeding the strongest baseline, T-Rex, by 9.1 percentage points. Our method performs particularly well on bottle-cap twisting, liquid transfer with a dropper, and zipper closing. As illustrated in Fig. 3a, these tasks require precise contact interactions under substantial visual occlusion, including establishing a secure grasp before twisting a cap, regulating pressure when squeezing and releasing a dropper, and grasping and pulling a zipper slider. The relatively low scores of $\pi_{0.5}$ on these tasks highlight the limitations of relying on visual observations alone. Although FTP-1 improves upon $\pi_{0.5}$, it remains substantially behind our method on bottle-cap twisting (40.0% versus 67.0%), consistent with the value of refining actions using real-time tactile feedback.

T-Rex performs competitively across tasks, but its fixed refinement interval in our evaluation, with tactile updates every four action steps, limits flexibility across tasks with different feedback requirements. In our rollouts, this configuration leads to prolonged execution on tasks such as whiteboard wiping. TF^2-Steering allows the refinement frequency to be adjusted at inference time, enabling task-dependent trade-offs between tactile responsiveness and execution overhead.

ViTacFormer performs poorly across the evaluated tasks, suggesting that pretrained representations are valuable for learning multistep dexterous manipulation from limited task-specific demonstrations. Meanwhile, $\pi_{0.5}$ + tactile performs worse than $\pi_{0.5}$ in the real-world evaluation, with average progress success rate decreasing from 40.2% to 35.2%. We hypothesize that adding the composite tactile image introduces a distribution shift: the resulting four-image input differs from the pretraining configuration and may hinder the transfer of pretrained visual representations.

4.3 Ablation Studies

We examine the contributions of model design and tactile pretraining to the performance of TF^2-Steering.

Ablation on Model Design. We conduct real-world ablations along three dimensions: tactile modalities, tactile integration strategies, and architectural choices. As shown in Table 3, removing both tactile modalities reduces the average progress success rate from 72.3% to 61.7%. Using predicted tactile alone raises this score to 63.0%, while using real-time tactile alone yields a larger

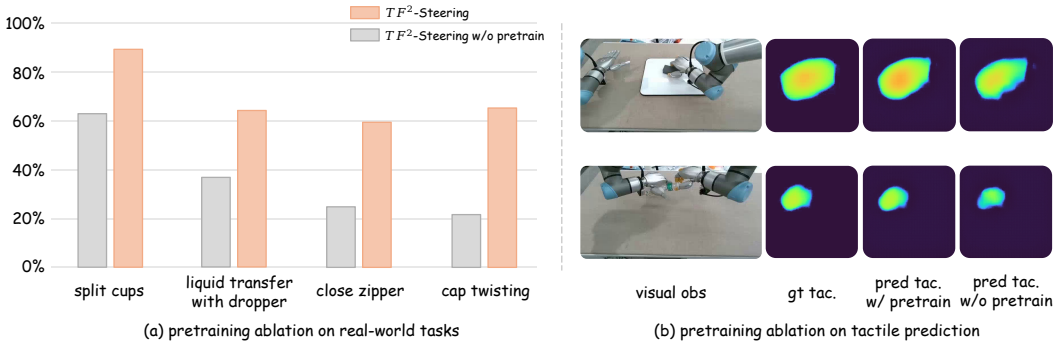

Figure 4: Effect of tactile pretraining. **(a)** Real-world progress success rates with and without tactile pretraining on four manipulation tasks.

improvement to 69.7%. Combining both modalities achieves the highest average performance, suggesting that tactile forecasts and real-time feedback provide complementary information for dexterous manipulation.

For tactile integration, action-space residual correction achieves an average score of 61.3%, comparable to the variant without tactile inputs (61.7%). This result suggests that, under our evaluation settings, correcting an already generated action sequence offers limited benefits compared with refining the flow-matching velocity throughout action generation.

At the architectural level, removing asynchronous inference reduces performance on bottle-cap twisting from 67.0% to 56.0%, supporting the value of frequent tactile-guided action updates for precise contact interactions. We also jointly ablate the attention mask and stop-gradient operation between the action expert and tactile expert. The attention mask prevents the action expert from accessing tactile features, while stop-gradient prevents the tactile correction loss from updating the action expert. Together, these constraints preserve the separation between base action generation and tactile-dependent delta-velocity refinement. Removing both reduces the average score to 62.7%, close to the variant without tactile inputs. This degradation may arise from the dominance of the action expert's pretrained visual representations during joint optimization, limiting the contribution of tactile information to action refinement. Further details of the model-design ablation configurations are provided in Appendix A.3.1.

Ablation on Pretraining. Our three-stage training procedure adapts a pretrained world-action model to tactile-guided manipulation through tactile prediction pretraining, tactile velocity steering pretraining, and task-specific post-training. Tactile pretraining initializes the tactile world model's ability to forecast future tactile observations and teaches the tactile expert to use predicted and real-time tactile information to produce delta-velocity corrections. As shown in Fig. 4a, tactile pretraining improves progress success rates across all four evaluated tasks, demonstrating its value beyond task-specific post-training alone. The pretraining ablation configuration is detailed in Appendix A.3.2. Figure 4b further shows that training the tactile world model only on task-specific post-training data produces poor tactile forecasts. This qualitative comparison suggests that the limited downstream data are insufficient to learn accurate future tactile prediction, motivating pretraining on a larger and more diverse tactile dataset.

5 Conclusion

We presented TF^2-Steering, a dual-rate framework that combines cached tactile forecasts and real-time feedback for contact-rich manipulation. A tactile expert predicts delta-velocity corrections to steer the action expert's flow-matching velocity during action generation. Simulation and real-world experiments, supported by ablations, demonstrate the effectiveness of this refinement strategy and the value of tactile pretraining.

AI USE STATEMENT

We used generative AI tools, including OpenAI Codex, to polish the manuscript's language and assist with code writing during implementation. The authors reviewed and revised the AI-assisted text and reviewed and tested the AI-assisted code. The authors take full responsibility for the final manuscript and implementation.

ETHICS STATEMENT

This work investigates contact-rich robotic manipulation in simulation and on a laboratory tabletop platform. A central ethical consideration is the risk of injury or property damage from unintended robot contact. The reported task performance does not establish safety guarantees for operation around people or in untested environments. Deployment beyond the evaluated settings would require additional risk assessment, appropriate hardware and software safeguards, and human oversight. Reuse or release of demonstration data should respect the applicable data licenses and protect the privacy of teleoperators and bystanders.

REPRODUCIBILITY STATEMENT

Section 3 describes the model architecture, velocity-field steering procedure, and training objectives. Section 4 specifies the evaluated tasks, baselines, and metrics. Appendix A.1 documents the pre-training data mixtures, action and state representations, and optimization settings. Appendix A.2 provides the robot and sensor setup, demonstration collection procedure, baseline adaptation details, and evaluation protocols, including the task-specific progress-scoring rules. Appendix A.3 specifies the ablation configurations. Representative execution sequences are provided in Appendix A.4, with full executions in the supplementary videos.

REFERENCES

Michal Balcerak, Tamaz Amiranashvili, Antonio Terpin, Suprosanna Shit, Lea Bogensperger, Sebastian Kaltenbach, Petros Koumoutsakos, and Bjoern Menze. Energy matching: Unifying flow matching and energy-based models for generative modeling. *Advances in Neural Information Processing Systems*, 38:8583–8609, 2026.

Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.

Rui Cai, Jun Guo, Xinze He, Piaopiao Jin, Jie Li, Bingxuan Lin, Futeng Liu, Wei Liu, Fei Ma, Kun Ma, et al. Xiaomi-robotics-0: An open-sourced vision-language-action model with real-time execution. *arXiv preprint arXiv:2602.12684*, 2026.

Baijun Chen, Weijie Wan, Tianxing Chen, Xianda Guo, Congsheng Xu, Yuanyang Qi, Haojie Zhang, Longyan Wu, Tianling Xu, Zixuan Li, et al. Univtac: A unified simulation platform for visuo-tactile manipulation data generation, learning, and benchmarking. *arXiv preprint arXiv:2602.10093*, 2026a.

Wendi Chen, Han Xue, Yi Wang, Fangyuan Zhou, Jun Lv, Yang Jin, Shirun Tang, Chuan Wen, and Cewu Lu. Implicitrdp: An end-to-end visual-force diffusion policy with structural slow-fast learning. *IEEE Robotics and Automation Letters*, 2026b.

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 and Automation Letters*, 2026.

Xuhui Fan, Hongyu Wu, Longbing Cao, et al. Scot: Unifying consistency models and rectified flows via straight-consistent trajectories. *Advances in Neural Information Processing Systems*, 38: 141438–141465, 2026.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Yankai Fu, Ning Chen, Junkai Zhao, Heng Zhang, Guocai Yao, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. Decal: Towards physically-grounded dexterous vision-language-action models via contact-aware latent co-imagination. *arXiv preprint arXiv:2609.09119*, 2026.

Abraham George, Selam Gano, Pranav Katragadda, and Amir Barati Farimani. Vital pretraining: Visuo-tactile pretraining for tactile and non-tactile manipulation policies. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 258–264. IEEE, 2025.

Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, et al. Ltx-video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024.

Peng Hao, Chaofan Zhang, Dingzhe Li, Xiaoge Cao, Xiaoshuai Hao, Shaowei Cui, and Shuo Wang. Tla: Tactile-language-action model for contact-rich manipulation. *arXiv preprint arXiv:2503.08548*, 2025.

Zihao He, Hongjie Fang, Jingjing Chen, Hao-Shu Fang, and Cewu Lu. Foar: Force-aware reactive policy for contact-rich robotic manipulation. *IEEE Robotics and Automation Letters*, 10(6): 5625–5632, 2025.

Liang Heng, Haoran Geng, Kaifeng Zhang, Pieter Abbeel, and Jitendra Malik. Vitacformer: Learning cross-modal representation for visuo-tactile dexterous manipulation. *arXiv preprint arXiv:2506.15953*, 2025.

Chengkai Hou, Kun Wu, Jiaming Liu, Zhengping Che, Di Wu, Fei Liao, Guangrun Li, Jingyang He, Qiuxuan Feng, Zhao Jin, et al. Robomind 2.0: A multimodal, bimanual mobile manipulation dataset for generalizable embodied intelligence. *arXiv preprint arXiv:2512.24653*, 2025.

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. *arXiv preprint arXiv:2507.09160*, 2025.

Yuzhe Huang, Pei Lin, Wanlin Li, Daohan Li, Jiajun Li, Jiaming Jiang, Chenxi Xiao, and Ziyuan Jiao. Taf-vla: Tactile-force alignment in vision-language-action models for force-aware manipulation. *arXiv preprint arXiv:2601.20321*, 2026.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Vladimir Kulikov, Matan Kleiner, Inbar Huberman-Spiegelglas, and Tomer Michaeli. Flowedit: Inversion-free text-based editing using pre-trained flow models. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19721–19730. IEEE, 2025.

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. *arXiv preprint arXiv:2605.07308*, 2026a.

Yang Li, Zhaxizhuoma Zhaxizhuoma, Hongru Jiang, Junjie Xia, Hongquan Zhang, Jinda Du, Yunsong Zhou, Jia Zeng, Ce Hao, Jieji Ren, et al. Forcevla2: Unleashing hybrid force-position control with force awareness for contact-rich manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8911–8920, 2026b.

Jason Jingzhou Liu, Yulong Li, Kenneth Shaw, Tony Tao, Ruslan Salakhutdinov, and Deepak Pathak. Factr: Force-attending curriculum training for contact-rich policy learning. *arXiv preprint arXiv:2502.17432*, 2025.

Yunfan Lou, Yifan Ye, Yankai Fu, Jun Cen, Xiaowei Chi, Yaoxu Lyu, Peidong Jia, Sirui Han, Zhihe Lu, and Shanghang Zhang. Dream-tac: A unified tactile world action model for contact-rich robot manipulation. *arXiv preprint arXiv:2606.08737*, 2026.

Haoxiang Ma, Junhao Cai, Xiaoxu Xu, Hao Li, Yuyin Yang, Yang Tian, Jiafei Cao, Hongrui Zhu, Zherui Qiu, Yuqiang Yang, et al. Internvla-a1. 5: Unifying understanding, latent foresight, and action for compositional generalization. *arXiv preprint arXiv:2607.04988*, 2026a.

Zipei Ma, Xiaofei Wei, Junzhe Jiang, Shunlin Lu, and Li Zhang. Tacpac: Tactile prediction and real-time action correction in world-action models for contact-rich manipulation. *arXiv preprint arXiv:2609.05266*, 2026b.

Dantong Niu, Zhuoyang Liu, Zekai Wang, Boning Shao, Zhao-Heng Yin, Anirudh Pai, Yuvan Sharma, Stefano Saravalle, Ruijie Zheng, Jing Wang, et al. T-rex: Tactile-reactive dexterous manipulation. *arXiv preprint arXiv:2606.17055*, 2026.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

Dogyun Park, Sojin Lee, Sihyeon Kim, Taehoon Lee, Youngjoon Hong, and Hyunwoo J Kim. Constant acceleration flow. *Advances in Neural Information Processing Systems*, 37:90030–90060, 2024.

Maitreya Patel, Song Wen, Dimitris N Metaxas, and Yezhou Yang. Flowchef: Steering of rectified flow models for controlled generations. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15308–15318. IEEE, 2025.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Md Shahriar Rahim Siddiqui, Moshe Eliasof, and Eldad Haber. Graph flow matching: Enhancing image generation with neighbor-aware flow fields. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 25463–25471, 2026.

NeoteAI Team, Fudan TEAI Team, et al. n_0-vtla: Scaling vision-tactile-language-action model with latent tactile tokens. *arXiv preprint arXiv:2607.23782*, 2026.

Shuai Tian, Yupeng Zheng, Yuhang Zheng, Songen Gu, Yujie Zang, Yuxing Qin, Weize Li, Haoran Li, Wenchao Ding, and Dongbin Zhao. Vt-wam: Visual-tactile world action model for contact-rich manipulation. *arXiv preprint arXiv:2607.02503*, 2026.

Yi Wang, Renjun Wu, Jinyan Liu, and Xuesong Li. Vitar: Visuo-tactile residual adaptation for foundation vla manipulation. *arXiv preprint arXiv:2608.15816*, 2026.

Shihan Wu, Xuecheng Liu, Shaoxuan Xie, Pengwei Wang, Xinghang Li, Bowen Yang, Zhe Li, Kai Zhu, Hongyu Wu, Yiheng Liu, et al. Robocoin: An open-sourced bimanual robotic data collection for integrated manipulation. *arXiv preprint arXiv:2511.17441*, 2025.

Siyu Wu, Linjing You, Junjie Zhu, Yaozu Liu, Huang Kaixiang, Chen Yonghang, Jituo Li, Changhao Zhang, Jian Liu, Zhu Hengshuo, et al. Tactile-wam: Touch-aware world action model with tactile asymmetric attention. *arXiv preprint arXiv:2606.26663*, 2026.

Chenxi Xie, Minghan Li, Shuai Li, Yuhui Wu, Qiaosi Yi, and Lei Zhang. Dnaedit: Direct noise alignment for text-guided rectified flow editing. *Advances in Neural Information Processing Systems*, 38:124456–124474, 2026.

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. *arXiv preprint arXiv:2503.02881*, 2025.

Jiawen Yu, Hairuo Liu, Qiaojun Yu, Jieji Ren, Ce Hao, Haitong Ding, Guangyu Huang, Guofan Huang, Yan Song, Panpan Cai, et al. Forcevla: Enhancing vla models with a force-aware moe for contact-rich manipulation. *Advances in Neural Information Processing Systems*, 38:93409–93439, 2026.

Chengbo Yuan, Zicheng Zhang, Mingjie Zhou, Wendi Chen, Yi Wang, Zhuoyang Liu, Dantong Niu, Shuo Wang, Hui Zhang, Wenkang Zhang, et al. Ftp-1: A generalist foundation tactile policy across tactile sensors for contact-rich manipulation. *arXiv preprint arXiv:2606.13102*, 2026a.

Haoran Yuan, Weigang Yi, Zhenyu Zhang, Wendi Chen, Yuchen Mo, Jiashi Yin, Xinzhuo Li, Xiangyu Zeng, Chuan Wen, Cewu Lu, et al. Vtam: Video-tactile-action models for complex physical interaction beyond vlas. *arXiv preprint arXiv:2603.23481*, 2026b.

Chaofan Zhang, Peng Hao, Xiaoge Cao, Xiaoshuai Hao, Shaowei Cui, and Shuo Wang. Vtla: Vision-tactile-language-action model with preference learning for insertion manipulation. *Biomimetic Intelligence and Robotics*, pp. 100333, 2026a.

He Zhang, Lingzhu Xiang, Haitao Lin, Zeyu Huang, Minghui Wang, Dingyan Zhong, Yubo Dong, Yihao Wu, Yongming Rao, Dongsheng Zhang, et al. Hy-embodied-0.5-vla: From vision-language-action models to a real-world robot learning stack. *arXiv preprint arXiv:2606.14409*, 2026b.

Kaidi Zhang, Heng Zhang, Zhengtong Xu, Zhiyuan Zhang, Md Rakibul Islam Prince, Xiang Li, Xiaojing Han, Yuhao Zhou, Arash Ajoudani, and Yu She. Tacvla: Contact-aware tactile fusion for robust vision-language-action manipulation. *arXiv preprint arXiv:2603.12665*, 2026c.

Qinglun Zhang, Zhen Liu, Haoqiang Fan, Guanghui Liu, Bing Zeng, and Shuaicheng Liu. Flowpolicy: Enabling fast and robust 3d flow-based policy via consistency flow matching for robot manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 14754–14762, 2025.

Yike Zhang, Yaonan Wang, Xinxin Sun, Kaizhen Huang, Zhiyuan Xu, Junjie Ji, Zhengping Che, Jian Tang, and Jingtao Sun. Craft: Adapting vla models to contact-rich manipulation via force-aware curriculum fine-tuning. *arXiv preprint arXiv:2602.12532*, 2026d.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

Wenliang Zhao, Minglei Shi, Xumin Yu, Jie Zhou, and Jiwen Lu. Flowturbo: Towards real-time flow-based image generation with velocity refiner. *Advances in Neural Information Processing Systems*, 37:4148–4176, 2024.

Yuhang Zheng, Songen Gu, Weize Li, Yupeng Zheng, Yujie Zang, Shuai Tian, Xiang Li, Ce Hao, Chen Gao, Si Liu, et al. Omnivta: Visuo-tactile world modeling for contact-rich robotic manipulation. *arXiv preprint arXiv:2603.19201*, 2026.

Jianbo Zhou, Boyuan Zhao, Yuzheng Zhang, Yiyang Chen, Wenxin Chen, Qiuyue Li, Xiangyang Gu, Yuhan Cao, Xiao Xia, Yanzhe Hu, et al. Tacforcing: Streaming action generation with execution-time tactile feedback. *arXiv preprint arXiv:2608.25798*, 2026a.

Jianyi Zhou, Feiyang Hong, Yunhao Li, Yicheng Zhao, Yongjue Cen, Zirui Liu, Jiakang Huang, Zirui Chen, Ruiyang Zhang, Weizhuo Zhu, et al. Touchworld: A predictive and reactive tactile foundation model for dexterous manipulation. *arXiv preprint arXiv:2607.07287*, 2026b.

A Appendix

A.1 Pretraining Details

A.1.1 World-Action Backbone Pretraining

Data and training setup. We pretrain the world-action backbone end-to-end on a mixture of AgiBot World (Bu et al., 2025), RoboMIND (Hou et al., 2025), Open X-Embodiment (O'Neill et al., 2024), and RoboCOIN (Wu et al., 2025). The video world model, modality compression Q-Former, and action expert are jointly optimized for visual prediction and action generation. This stage establishes the backbone before tactile-specific pretraining. We use the action annotations supplied by each dataset, which may contain joint-space commands, end-effector (EEF) commands, or both. These annotations are mapped to a shared 128-dimensional representation described below.

Visual observations are resized and center-cropped to 192×256 pixels. Future video frames are sampled at a target rate of 15 frames per second, with 24 future frames per training sample. The action expert predicts a sequence of 50 actions conditioned on visual features and the robot state. This horizon specifies backbone pretraining; the execution blocks used for downstream tactile refinement are defined separately in the main method. Action sequences retain the source datasets' sampling rates, without a common action-rate resampling target. Normalization statistics are maintained by dataset identifier.

We train for $100{,}000$ optimizer steps with a global batch size of 512. We use AdamW with a base learning rate of 10^{-5} and an action-expert learning rate of 10^{-4}, followed by cosine decay with a minimum learning rate of 10^{-6} after $1{,}000$ warmup steps. Delay augmentation is applied with probability 0.5, with uniformly sampled delays up to 25 action steps. Table 4 summarizes the main configuration.

Table 4: **World-action backbone pretraining configuration.** The training run uses $100{,}000$ optimizer steps and a global batch size of 512. The action horizon and delay settings refer to backbone pretraining.

Setting	Value
Inputs and architecture	
Image resolution ($H \times W$)	192×256
Future video frames / target frame rate	24 / 15 fps
Action horizon	50 steps
Action / state dimension	128 / 128
Video transformer (layers / heads / head dimension)	28 / 32 / 64
Multi-view cross-attention	Enabled
Action transformer (layers / hidden dimension / dropout)	16 / 1024 / 0.2
Optimization	
Optimizer / (β_1, β_2)	AdamW / $(0.9, 0.95)$
Training steps / global batch size	100,000 / 512
Base / action-expert learning rate	10^{-5} / 10^{-4}
Learning-rate schedule	Cosine decay; minimum 10^{-6}
Warmup steps / random seed	1,000 / 42
AdamW ϵ / weight decay	10^{-8} / 10^{-8}
Gradient clipping threshold / accumulation steps	1.0 / 1
Regularization and augmentation	
State / caption dropout probability	0.1 / 0.06
Delay augmentation probability	0.5
Maximum delay / sampling distribution	25 action steps / uniform
Temporal scaling / mirror augmentation	Disabled / disabled

Canonical action and state representation. To accommodate different embodiments and annotation formats, we use a fixed 128-dimensional layout for both actions and proprioceptive states. Table 5 gives the slot allocation using zero-based, inclusive indices. Each group has a fixed capacity; only coordinates supported by the dataset and embodiment are populated. Joint-space and EEF annotations occupy separate groups, so datasets providing both can retain both representations in the same vector.

TCP groups are populated when the dataset supplies EEF annotations and are otherwise zero-filled and masked out. State and action validity masks are constructed separately according to their available annotations.

Table 5: **Canonical 128-dimensional action and state layout.** Indices are zero-based and inclusive. Group dimensions indicate slot capacity, rather than the number of active degrees of freedom for every embodiment. TCP denotes the tool center point.

Indices	Dimension	Assigned quantity
0–9	10	Left-arm joint representation
10–19	10	Left TCP / end-effector representation
20	1	Left gripper
21–49	29	Left dexterous hand
50–59	10	Right-arm joint representation
60–69	10	Right TCP / end-effector representation
70	1	Right gripper
71–99	29	Right dexterous hand
100–127	28	Reserved; always inactive

Unused coordinates, including unsupported body parts and reserved slots, are zero-filled and marked invalid. For example, an arm with seven supplied joint coordinates occupies seven entries of its ten-dimensional joint group; the remaining three entries are inactive. After normalization, inactive coordinates are reset to zero, and the data pipeline supplies explicit state and action masks alongside the corresponding tensors. The canonical layout therefore preserves the distinction between an unavailable quantity and a valid coordinate whose value happens to be zero.

A.1.2 Tactile Pretraining

Data mixture. Both tactile pretraining stages use OpenNeoData (Team et al., 2026), the T-REX dataset (Niu et al., 2026), and FTP-1-Sharpa (Yuan et al., 2026a), with sampling weights of 30%, 40%, and 30%, respectively. The first stage uses this tactile-data mixture exclusively. In the second stage, tactile data account for 90% of the training mixture, retaining the same relative weights across the three sources. The remaining 10% consists of tactile-free data from world-action backbone pretraining and supplements training of the visual and base-action pathways. Visual inputs use a resolution of 192×256 pixels, and tactile prediction targets span 24 future frames at 15 frames per second. Table 6 summarizes the key settings.

Stage 1: Tactile world model pretraining. We initialize the visual conditioning stream from the pretrained world-action backbone and keep the LTX-Video model frozen. Only the tactile world model is optimized, using a flow-matching prediction objective in the shared VAE latent space. Visual features provide conditioning through the unidirectional attention pathway, with gradients stopped at the visual stream. The video prediction loss is disabled in this stage. We sample tactile noise levels using a logit-normal sampler with logit mean 0 and standard deviation 1, and train for $50{,}000$ optimizer steps with a global batch size of $1{,}024$.

Stage 2: Tactile correction pretraining. We load the tactile world model weights learned in Stage 1 and initialize the video world model, Q-Former, and action expert from the pretrained world-action backbone described in Appendix A.1.1. In particular, the action expert retains its pretrained action-generation weights. We introduce the tactile expert with a zero-initialized delta-velocity output head, and jointly train all four modules and the Q-Former using the visual prediction, tactile prediction, base-action, and steering objectives defined in the main method. The unidirectional attention and gradient-isolation mechanisms remain in place during this joint optimization.

The tactile world model supplies its final-block hidden states to the tactile expert, which combines them with real-time tactile features to learn local corrections to the base flow-matching velocity. Tactile prediction training uses the noise endpoint $\sigma = 1$ in this stage, supporting single-step prediction. Actions are sampled at 30 Hz with a pretraining horizon of 50 steps. Real-time tactile observations are sampled at offsets of 0, 4, 8, or 12 action steps, and the steering loss is evaluated over a 10-step window following the sampled offset. Relative-time encoding identifies the position

of each action with respect to the feedback time. We train for 50,000 optimizer steps with a global batch size of 512.

Both stages use AdamW with $(\beta_1, \beta_2) = (0.9, 0.95)$ and $\epsilon = 10^{-8}$, gradient clipping at 1.0, and cosine learning-rate decay with a minimum learning rate of 10^{-6}.

Table 6: **Key settings for the two tactile pretraining stages.** Stage 1 trains only the tactile world model; Stage 2 jointly trains all four modules and the Q-Former. Batch sizes are global; dashes indicate inapplicable settings.

Setting	Stage 1	Stage 2
Training steps	50,000	50,000
Global batch size	1,024	512
Tactile / tactile-free sampling	100% / 0%	90% / 10%
Video model and Q-Former learning rate	–	10^{-5}
Tactile world model learning rate	10^{-4}	5×10^{-5}
Action expert learning rate	–	5×10^{-5}
Tactile expert learning rate	–	10^{-4}
Warmup steps	5,000	1,000
Weight decay	0	10^{-8}
Tactile noise sampling	Logit-normal	Endpoint ($\sigma = 1$)

A.2 Additional Experimental Details

A.2.1 Real-Robot Platform and Teleoperation

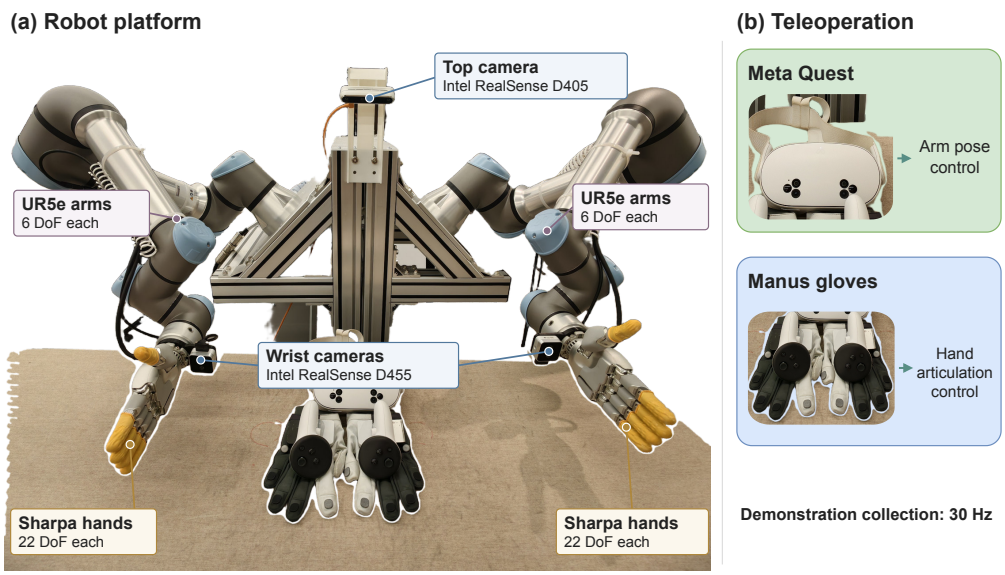

Figure 5: **Real-robot platform and demonstration collection. (a)** Two UR5e arms equipped with Sharpa dexterous hands share a tabletop workspace. A top-mounted Intel RealSense D405 and two wrist-mounted D455 cameras provide complementary visual observations. **(b)** Meta Quest supplies arm pose commands, while Manus gloves control hand articulation. Demonstrations are collected at 30 Hz.

Robot and sensing hardware. As shown in Fig. 5(a), our bimanual platform consists of two 6-DoF UR5e robotic arms, each equipped with a 22-DoF Sharpa dexterous hand. Both arms operate over a shared tabletop workspace. The visual observation system comprises three cameras: one Intel

RealSense D405 mounted above the workspace and one Intel RealSense D455 mounted at each wrist. The top camera provides an overall view of the scene, while the wrist cameras provide local views near the hands and manipulated objects. These visual observations complement the tactile feedback acquired during contact interactions.

Teleoperation interface. We collect demonstrations using Manus gloves and a Meta Quest system (Fig. 5(b)). The Manus gloves capture the operator's finger articulation to control the Sharpa hands, while tracked motion from the Meta Quest controllers provides pose commands for the UR5e arms. This interface allows the operator to coordinate arm motion and finger articulation during bimanual manipulation.

Demonstration acquisition. Demonstrations are collected at 30 Hz and contain multi-view visual observations, tactile observations, robot states, and action commands. We collect 100 demonstrations for each of the six real-world tasks described in the main text, yielding 600 demonstrations in total.

A.2.2 Simulation Training and Evaluation

We evaluate on UniVTAC (Chen et al., 2026a). For methods trained in our experiments, we partition the six tasks according to their visual inputs and train one checkpoint jointly on the tasks within each group. The single-view group consists of *Lift Bottle*, *Pull Out Key*, *Put Bottle*, and *Insert Hole*; the multi-view group consists of *Lift Can* and *Insert Tube*. Each method therefore uses two group-level checkpoints, with each checkpoint evaluated on all tasks in its corresponding group.

We follow the official UniVTAC evaluation protocol and task-success criteria. Each method evaluated by us is tested for 50 trials per task. The task-level success rate is the fraction of successful trials, expressed as a percentage. The overall score is the unweighted mean of the six task-level success rates. The ACT and VITaL scores in Table 1 are reproduced from UniVTAC. The two-checkpoint training setup and 50-trial evaluation described here apply to methods trained and evaluated by us, rather than these published reference results.

A.2.3 Real-World Training and Baseline Adaptation

Training data and baseline fine-tuning. Each real-world task provides 100 demonstrations collected at 30 Hz using the platform described in Appendix A.2.1. For T-Rex (Niu et al., 2026) and FTP-1 (Yuan et al., 2026a), we follow their respective official training procedures and perform 20,000 task-specific optimization steps on the collected demonstrations.

Adapting the $\pi_{0.5}$ action interface. The released $\pi_{0.5}$ checkpoint used in our experiments has a 32-dimensional action interface, which is insufficient to accommodate the arm and dexterous-hand commands of our UR5e–Sharpa platform. We expand its linear action projection to support an 80-dimensional action representation. TCP commands retain their assigned positions within the original first 32 dimensions, while the Sharpa hand-joint commands are assigned to the newly added dimensions. With zero-based indexing, the original action subspace occupies indices 0–31 and the added hand-command subspace occupies indices 32–79. This 80-dimensional interface is specific to the $\pi_{0.5}$ adaptation; the canonical layout used by our backbone is described in Table 5.

A.2.4 Real-World Evaluation and Progress Scoring

We evaluate each method for 20 trials on each real-world task. Each trial receives a normalized progress score $p_i \in [0, 1]$ according to the task-specific milestones in Table 7. For sequential tasks, the listed values are cumulative scores: a trial receives the score of the furthest completed milestone, and scores are not summed across milestones. A trial that reaches none of the listed milestones receives 0. For fruit placement, the score increases by 0.25 for each of the four objects successfully placed on the plate. The progress success rate (PSR) for a task is

$$\mathrm{PSR} = \frac{100}{N} \sum_{i=1}^{N} p_i, \qquad N = 20. \tag{7}$$

We report PSR as a percentage, and compute the overall real-world score as the unweighted mean of the six task-level PSRs.

Table 7: **Progress scoring for real-world evaluation.** Scores range from 0 to 1 for each trial. Sequential-task scores indicate cumulative progress, rather than additive rewards.

Task	Progress milestones and scores
Fruit on Plate	0.25 for each object successfully placed on the plate; 1.0 when all four objects are placed.
Close Zipper	0.2: the left hand lifts the bag; 0.6: the right hand begins pulling the zipper, which is not yet fully closed; 1.0: the zipper is fully closed.
Wipe Whiteboard	0.2: the robot grasps the eraser; 0.6: it performs an initial wiping stroke on the board; 1.0: the board is wiped clean.
Twist Cap	0.2: the left hand lifts the bottle; 0.6: the robot successfully rotates the cap; 1.0: the cap is fully unscrewed.
Split Cups	0.2: the robot lifts the cups; 0.6: it inserts its fingers into a cup; 1.0: one cup is separated from the others.
Dropper Transfer	0.2: the robot grasps the dropper; 0.8: liquid is transferred into the beaker; 1.0: the dropper is put back after the transfer.

A.3 Ablation Details

A.3.1 Model Design Ablations

Initialization and post-training. All model-design variants in Table 3 are initialized from the same checkpoint obtained after Stage 2 tactile correction pretraining (Appendix A.1.2). We apply the corresponding architectural or objective changes before task-specific post-training and train each variant separately. The reported results therefore evaluate models adapted to their ablated configurations, rather than inference-time modifications of a fully post-trained model. We use the progress scoring protocol in Appendix A.2.4.

Without both tactile modalities. We remove all tactile-specific modules and computation paths, including the tactile world model, real-time tactile encoding, and tactile expert. The retained video world model, Q-Former, and action expert undergo world-action backbone post-training. Actions are generated entirely through the base action expert, without tactile prediction, feature fusion, or delta-velocity refinement.

Without predicted or real-time tactile. For *w/o predicted tactile*, we disable the predicted-tactile subexpert and retain the real-time tactile subexpert. For *w/o real-time tactile*, we disable the real-time tactile subexpert and retain the predicted-tactile subexpert. In both cases, we bypass sigmoid weighting and pass the active subexpert's feature directly to the subsequent tactile-expert computation. The remaining branch thus continues to support delta-velocity prediction without a weighted combination of the two tactile sources.

Action-space residual correction. We retain the same model architecture but change the correction target and the point at which the correction is applied. The action expert first completes flow-matching generation to produce the base action sequence. The tactile branch then predicts an action-space residual that corrects this final sequence. During post-training, we replace the velocity-steering loss with a residual loss on the final actions. This variant compares refinement during flow-matching generation with correction after base-action generation using the same architecture.

Without asynchronous inference. We set the training delay to $d = 0$ and remove the local correction-window restriction, evaluating the steering loss over all H actions in the chunk. At inference, the world-model planning system and the action-generation system operate at the same update frequency: world priors and action chunks are refreshed synchronously. This variant removes the separation between low-frequency planning and high-frequency tactile refinement during both post-training and execution.

Without attention masking and stop-gradient. We jointly remove the attention-mask and stop-gradient constraints between the action expert and tactile expert, and independently post-train this variant from the same Stage 2 checkpoint. Both the forward attention restriction and the gradient-

isolation mechanism are removed during post-training. This experiment evaluates their combined role in separating base action generation from tactile refinement.

A.3.2 Pretraining Ablation

The *w/o pretrain* variant in Fig. 4(a) uses the original world-action checkpoint obtained before either tactile pretraining stage (Appendix A.1.1). The video world model, Q-Former, and action expert retain these pretrained weights, while the tactile world model and tactile expert are randomly initialized. We skip both tactile world model pretraining and tactile correction pretraining, and proceed directly to task-specific post-training using the full model's joint objective.

Both tactile modules and both tactile sources remain available in this variant during post-training and inference. The comparison therefore evaluates the contribution of the two tactile pretraining stages on top of the original world-action initialization. The fully pretrained counterpart starts task-specific post-training from the checkpoint obtained after Stage 2.

A.4 Real-World Policy Rollouts

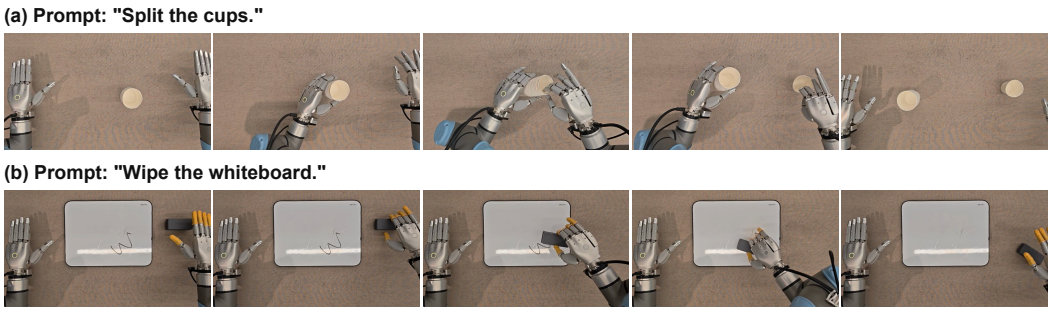

Figure 6: **Representative real-world policy rollouts. (a)** Split Cups. **(b)** Wipe Whiteboard. Each row shows five selected frames in chronological order from left to right, with the task prompt at the upper left.

Figure 6 illustrates the progression of two contact-rich tasks during policy execution. In *Split Cups*, the robot grasps the nested cups, engages them with the other hand, and separates a cup before placing the cups apart. In *Wipe Whiteboard*, the robot grasps the eraser, brings it into contact with the board, wipes across the marked region, and moves the eraser away after wiping. These selected frames complement the quantitative evaluations in the main text. Full executions are available in the supplementary videos.