SIMPLETOUCH: CAN VISION-LANGUAGE-ACTION MODELS MASTER CONTACT-RICH MANIPULATION WITHOUT TACTILE POLICY PRETRAINING?

Anonymous authors
Paper under double-blind review

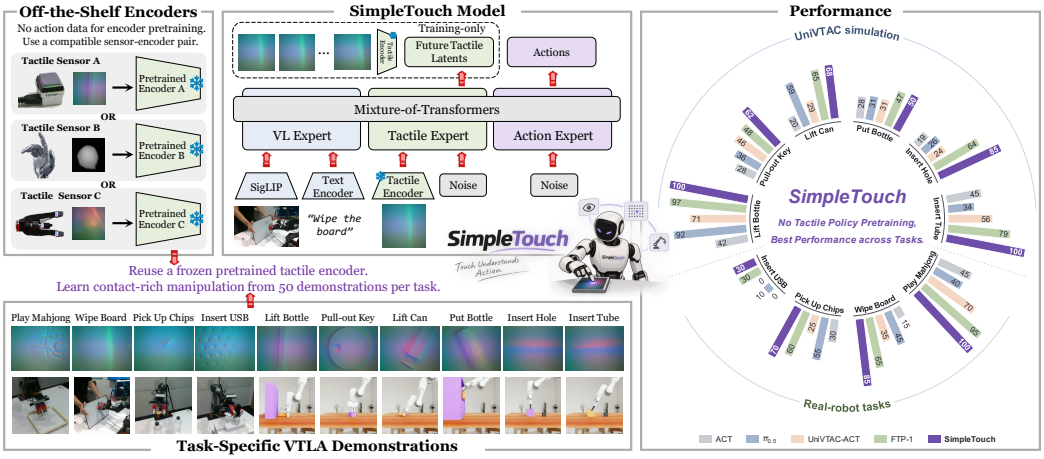

Figure 1: Overview of **SimpleTouch**. **SimpleTouch** combines an off-the-shelf pretrained tactile encoder with a tactile expert to learn from 50 task-specific demonstrations without tactile policy pretraining. The chart on the right reports success rates on six UniVTAC and four real-robot tasks.

ABSTRACT

Tactile sensing provides essential contact information for robotic manipulation, yet incorporating it into pretrained vision-language-action (VLA) models remains challenging. A common concern is that simply introducing touch during task-specific fine-tuning may fail to bridge the cross-modal gap, yielding limited gains or even reduced success. Consequently, existing methods often rely on large-scale tactile policy pretraining or separate visuotactile alignment, adding data requirements and training stages. We introduce **SimpleTouch**, a simple VLA extension that augments $\pi_{0.5}$ with a tactile expert, to test whether these additional stages are necessary. Leveraging all tokens from a frozen pretrained tactile encoder, the expert learns from action supervision and multi-horizon prediction of future tactile latents. This single-stage training uses only task demonstrations, without additional tactile policy pretraining or separate alignment. With 50 demonstrations per task, **SimpleTouch** achieves the highest success rate among evaluated methods on all six UniVTAC tasks. Its average success rate reaches 77.5%, compared with 45.2% for FTP-$\pi_{0.5}$ and 66.7% for FTP-1, corresponding to gains of 32.3 and 10.8 percentage points, respectively. Across four real-world tasks, it averages 71.3%, exceeding FTP-1 by 8.8 percentage points. These results demonstrate that, given pretrained VLA and tactile representations, additional tactile policy pretraining is not a prerequisite for strong performance on these tasks, offering a simpler route to contact-rich manipulation.

1 INTRODUCTION

Robot manipulation has made substantial progress through learned policies (Zhao et al., 2023; Chi et al., 2023) and pretrained vision-language-action (VLA) models (Kim et al., 2025b; Black et al.,

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.

2024). Much of this progress has relied on visual perception as the primary interface for scene understanding. However, visual observations do not fully determine contact-rich interaction: two manipulation states can appear visually similar while requiring different actions because their contact geometry, deformation, or slip state differs (Li et al., 2014; Yuan et al., 2017). Tactile sensing supplies local feedback about these otherwise hidden contact states, motivating policies that learn from touch and use it during manipulation (Xue et al., 2025; Huang et al., 2025; Lou et al., 2026).

However, adding touch to a pretrained policy does not automatically make it useful. Tactile observations lie outside the pretraining modalities of many VLA backbones, and directly incorporating them during fine-tuning can yield limited gains or even reduce success (Niu et al., 2026a; Li et al., 2026c; Zhang et al., 2026a). The central challenge is therefore not merely obtaining tactile features, but presenting them to a pretrained VLA in a form that preserves contact information while allowing the policy to learn their task-specific use from limited demonstrations. Representative responses include large-scale tactile policy pretraining, as in FTP-1 (Yuan et al., 2026a), and separate visuotactile representation alignment, as in VITaL (George et al., 2025). These routes provide useful priors but introduce additional data requirements and training stages. Meanwhile, pretrained tactile encoders already offer reusable contact representations (Zhao et al., 2025). This distinction between acquiring tactile representations and learning a tactile policy motivates our question: *given a pretrained VLA and tactile encoder, can a policy learn strong contact-rich manipulation from task-specific tactile demonstrations without additional tactile policy pretraining or a separate alignment stage?*

We introduce **SimpleTouch**, a simple recipe that augments $\pi_{0.5}$ (Physical Intelligence et al., 2025) with one tactile expert, without additional tactile policy pretraining or a separate alignment stage. Its design follows two principles. First, it preserves the pretrained tactile representation: we freeze the tactile encoder and retain all current tokens, including global and spatial features, as a stable source of contact information. Second, it learns the task-specific use of that representation through a tactile expert. The tactile expert processes current tactile features separately from the visual-language backbone, and the action expert reads them through layerwise attention. During joint training, the expert learns from both action supervision and multi-horizon prediction of future tactile latents through latent flow matching (Lipman et al., 2023). This task-trained pathway gives action generation direct access to high-fidelity tactile features. During task-level training, we jointly optimize the pretrained $\pi_{0.5}$ policy, including its visual-language backbone and action expert, together with the tactile expert, while keeping the tactile encoder frozen.

With 50 demonstrations per task, **SimpleTouch** achieves the highest success rate among the evaluated methods on all six UniVTAC (Chen et al., 2026a) tasks. Its mean success rate of 77.5% exceeds those of FTP-$\pi_{0.5}$ and the pretrained FTP-1 policy by 32.3 and 10.8 percentage points, respectively. Across four real-world tasks, **SimpleTouch** averages 71.3%, surpassing FTP-1 by 8.8 points and showing improvements in real-world experiments as well. At deployment, the tactile expert processes the current observation once, and its layerwise cache is reused throughout action-flow integration; future tactile prediction is used only as training supervision. Together, these results show that, given strong pretrained VLA and tactile representations, a frozen tactile representation and task-trained tactile expert can achieve strong performance without additional tactile policy pretraining or explicit alignment.

2 Related Work

Generalist Policy Learning. Generalist policy learning converts broad visual and language priors into continuous robot control across diverse tasks. The π series couples pretrained vision-language backbones with flow-matching action experts (Black et al., 2024; Physical Intelligence et al., 2025). Other VLA systems investigate large-scale data, efficient fine-tuning, execution, generalization, and modular policy development (Open X-Embodiment Collaboration et al., 2023; Khazatsky et al., 2024; NVIDIA et al., 2025; Pertsch et al., 2025; Kim et al., 2025a; Cai et al., 2026; Chen et al., 2025; StarVLA Community, 2026). Complementary world-action models incorporate predictive visual or feature dynamics into action learning, including Video Prediction Policy, LingBot-VA, Motus, DreamZero, LaWAM, and Fast-WAM (Hu et al., 2024; Li et al., 2026a; Zhang et al., 2026b; Bi et al., 2025a; Ye et al., 2026; Chen et al., 2026b; Yuan et al., 2026b). These directions establish strong reusable architectures and priors for action generation, but their observation interfaces are primarily built around vision, language, and proprioception rather than local tactile contact feedback.

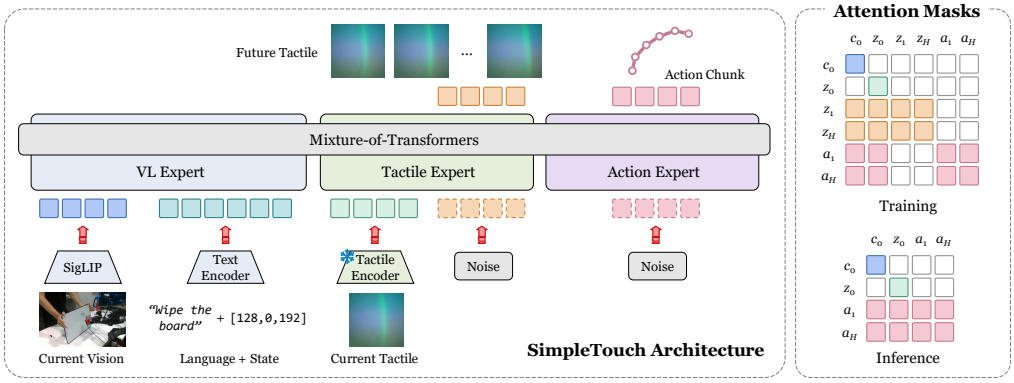

Figure 2: **SimpleTouch** architecture. A tactile expert uses complete current tactile features from a frozen encoder and learns multi-horizon future prediction. In the right-side training and inference masks, rows are queries, columns are keys/values, and colored cells mark permitted attention. The symbols c_0, z_0, $z_{1:H}$, and $a_{1:H}$ denote the current visual-language and state context, current tactile tokens, future tactile tokens, and action tokens, respectively, following the token order shown above.

Tactile Policy Learning. Tactile policy learning can be organized by where contact-sensitive priors are acquired. One route first learns how tactile observations participate in control through large-scale tactile policy pretraining or tactile-grounded mid-training, as in FTP-1, the N_0-TWAM/N_0-VTLA family, and T-Rex (Yuan et al., 2026a; NeoteAI Team & Fudan TEAI Team, 2026a;b; Niu et al., 2026a). A second route aligns tactile and visual representations before policy learning, including VITaL, UniTouch, and VT-MUSE (George et al., 2025; Yang et al., 2024; Xu et al., 2026). Both front-load resources beyond task-level learning. Other methods explore task-level tactile reasoning, adaptive fusion, or reactive feedback (Yu et al., 2023; Huang et al., 2024; Heng et al., 2025; Liu et al., 2025a; Huang et al., 2025; Bi et al., 2025b; Li et al., 2026c; Xue et al., 2025), while predictive approaches model future contact or multimodal dynamics (Lou et al., 2026; Niu et al., 2026b; Zheng et al., 2026). Across these settings, simply adding tactile input during fine-tuning does not consistently improve success (Yuan et al., 2026a; Niu et al., 2026a; Li et al., 2026c). An open question is whether strong contact-rich control instead requires an additional tactile policy-learning or alignment stage once pretrained VLA and tactile representations are available.

Pretrained Tactile Encoders. Pretrained tactile encoders address representation acquisition rather than action generation. Early work such as T-Dex learns self-supervised tactile embeddings from robotic play for downstream dexterous control (Guzey et al., 2023). Sparsh and Sparsh-X learn general-purpose representations from self-supervised vision-based and multisensory touch data, respectively (Higuera et al., 2024; 2025); the AnyTouch series models static and dynamic tactile signals across sensors and downstream tasks (Feng et al., 2025; 2026). T3 combines sensor-specific encoders with a shared Transformer trained across sensors and tasks (Zhao et al., 2025). Such encoders provide reusable contact-sensitive features across tasks and sensing configurations. This differs from tactile policy pretraining: encoders supply sensory representations, whereas policy pretraining also learns how touch influences action generation. Existing encoder work does not prescribe the downstream action interface or establish whether a separate tactile policy-training stage remains necessary. A detailed review of related work is provided in Appendix B.

3 Method

3.1 Problem Formulation

We study task-level learning of contact-rich manipulation from a pretrained vision-language-action (VLA) policy, a pretrained tactile encoder, and task-specific tactile demonstrations. The demonstrations form trajectories $\mathcal{D}$ of synchronized visual observations, language instructions, proprioceptive states, tactile observations, and actions. At environment step t, let $O_t = (I_t, \ell, s_t, U_t)$ denote these current observations, where I_t, ℓ, s_t, and U_t are visual observations, a language instruction, proprioceptive state, and tactile observations, respectively. The policy predicts an action chunk

$A_t = (a_t, \ldots, a_{t+H-1})$ over horizon H:

$$A_t \sim \pi_\Theta(\cdot \mid I_t, \ell, s_t, U_t). \tag{1}$$

Our goal is to learn this policy using $\mathcal{D}$ alone, without an additional tactile policy-pretraining stage or a separate visuotactile alignment stage. The trainable parameters are $\Theta = \{\theta_{\mathrm{VL}}, \theta_A, \theta_T\}$ for the visual-language backbone, action expert, and tactile expert, respectively; the tactile encoder parameters ϕ are held fixed during task-level training. Each trajectory also contains future tactile observations, which provide training-only supervision for the tactile expert.

Figure 2 summarizes the design. **SimpleTouch** preserves the complete current representation of a pretrained tactile encoder, routes it through a tactile expert, and uses future tactile prediction as a training-time objective for learning the tactile-action pathway. The pretrained VLA policy continues to supply visual-language and action priors; the tactile expert provides the task-trained interface from touch to action. Implementation details of **SimpleTouch** are provided in Appendix A.

3.2 Preserving Pretrained Tactile Features

The first design principle is to preserve, rather than re-learn, the pretrained tactile representation. We use pretrained T3 (Zhao et al., 2025) as a tactile encoder E_ϕ and keep its weights frozen. For the current tactile observation, it produces

$$Z_t = E_\phi(U_t). \tag{2}$$

We retain every global and spatial token in Z_t without pooling or learned compression, thereby preserving the spatial layout encoded by T3. Thus, the downstream policy can access the encoder's full current representation of contact, while task-level training only learns how that representation should be used for control. Future tactile targets are treated differently: they are pooled only for predictive supervision, not for the current tactile context used by the policy.

3.3 A Tactile Expert as a Tactile–Action Interface

The $\pi_{0.5}$ policy (Physical Intelligence et al., 2025) comprises a visual-language backbone and an action expert. We augment it with a tactile expert of the same Transformer architecture as the action expert but with independent parameters. Rather than inserting tactile tokens into the visual-language backbone, its current-touch computation $T_{\theta_T}^{\mathrm{current}}$ processes Z_t and produces the layerwise key-value cache

$$\mathcal{K}_t = \{(K_{T,t}^j, V_{T,t}^j)\}_{j=1}^L = T_{\theta_T}^{\mathrm{current}}(Z_t), \tag{3}$$

where L is the number of Transformer layers and j indexes them. Let $C_t = B_{\theta_{\mathrm{VL}}}(I_t, \ell, s_t)$ be the visual-language and state context produced by the pretrained backbone. At each action-expert layer, action queries read visual-language context, current tactile context, and noisy action tokens:

$$R_A^j = \mathrm{Attn}\Big(Q_A^j, [K_C^j; K_{T,t}^j; K_A^j], [V_C^j; V_{T,t}^j; V_A^j]\Big). \tag{4}$$

Here Attn is multi-head attention, and Q_A^j and R_A^j are the action-block queries and outputs. For $X \in \{C, T, A\}$, K_X^j, V_X^j are the layer-j projections of visual-language context, current tactile features, and noisy action tokens, respectively; semicolons denote token concatenation. Expert-specific projections make the tactile keys and values compatible with the action expert. This layerwise interface exposes the full tactile token sequence throughout action generation while allowing tactile processing to use its own parameters. The tactile expert consequently learns a task-specific transformation from pretrained tactile features to action-relevant context.

3.4 Learning Contact Dynamics from Future Touch

Action supervision alone trains the tactile expert only through the downstream behavior objective. We add multi-horizon future tactile prediction as dense supervision of contact evolution. For offsets Δ within the action horizon, future tactile observations are encoded by the same encoder and form the target sequence

$$Y_t = \big[\mathcal{S}\big(\mathcal{P}(E_\phi(U_{t+\delta}))\big)\big]_{\delta \in \Delta}. \tag{5}$$

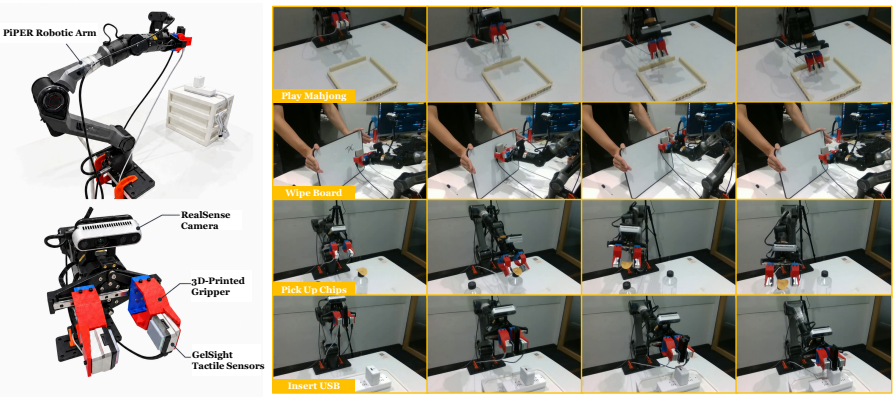

Figure 3: Real-world setup and rollouts. The left panel shows the PiPER platform and sensing hardware; the right panel shows all four tasks—Play Mahjong, Wipe Board, Pick Up Chips, and Insert USB—with time progressing from left to right throughout each rollout.

Here $\mathcal{P}$ retains global tokens and spatially pools patch tokens, while $\mathcal{S}$ applies fixed feature scaling. This asymmetry preserves all tokens for current control while keeping the multi-horizon prediction target tractable.

Following the Fast formulation of Fast-WAM (Yuan et al., 2026b), we treat future touch as an auxiliary prediction target rather than an intermediate condition for action generation:

$$\underbrace{p_\Theta(A_t, Y_t \mid O_t)}_{\text{task-level training}} = \underbrace{\pi_\Theta(A_t \mid O_t)}_{\text{action generation}} \underbrace{p_\Theta(Y_t \mid O_t)}_{\text{future-touch prediction}} . \tag{6}$$

Unlike future-conditioned formulations $p(Y_t \mid O_t)\, p(A_t \mid O_t, Y_t)$, the action factor depends only on current observations. We implement this separation with asymmetric attention: current tactile queries read current touch; future tactile queries read visual-language context, current touch, and noisy future tactile tokens; and action queries read current context but not future tactile tokens.

Both action chunks and future tactile targets are learned with conditional flow matching (Lipman et al., 2023). The joint objective is

$$\mathcal{L} = \mathcal{L}_{\text{act}} + \lambda_T \mathcal{L}_{\text{tac}}, \tag{7}$$

where both terms are flow-matching velocity losses and λ_T weights tactile prediction. The current and future paths share tactile-expert parameters, so $\mathcal{L}_{\text{tac}}$ trains the same expert that supplies current tactile context for action generation. We jointly optimize $\Theta = \{\theta_{\text{VL}}, \theta_A, \theta_T\}$.

At deployment, the tactile expert processes current touch once, and its layerwise cache $\mathcal{K}_t$ is reused throughout action-flow integration alongside the visual-language context. The auxiliary future-prediction branch is omitted, leaving action generation conditioned only on observed context. Appendix A.2 provides the complete token construction, attention graph, flow-matching objectives, and inference procedure.

4 Experiments

We organize our experiments around four questions. **First, how effective is SimpleTouch?** We benchmark it against seven strong baselines in simulation and four on the real robot, including the pretrained tactile policy FTP-1 (Section 4.2). **Second, what makes SimpleTouch effective?** We use controlled ablations to examine its two central principles: preserving pretrained tactile representations and learning their task-specific use through predictive supervision (Section 4.3). **Third, how data-efficient is task-level tactile learning?** We evaluate performance across different demonstration budgets to determine whether the method remains effective when task data are scarce (Section 4.4). **Finally, does predictive tactile learning require additional inference-time generation?** We compare alternative ways of coupling tactile prediction with action generation and evaluate both task performance and computational efficiency (Section 4.5).

Table 1: Results on the UniVTAC simulation benchmark. We report success rates (%). "Avg. w/o Lifts" excludes Lift Bottle and Lift Can. We highlight the **best** and second-best results.

Method	Lift Bottle	Pull-out Key	Lift Can	Put Bottle	Insert Hole	Insert Tube	Avg.	Avg. w/o Lifts
ACT	42	28	20	28	19	45	30.3	30.0
$\pi_{0.5}$	92	36	59	31	26	34	46.3	31.8
VITaL	72	47	8	32	25	34	36.3	34.5
UniVTAC-ACT	71	46	29	31	24	56	42.8	39.3
Tactile-VLA	97	32	15	10	41	56	41.8	34.8
FTP-$\pi_{0.5}$	77	30	26	19	47	72	45.2	42.0
FTP-1	97	48	65	47	64	79	66.7	59.5
SimpleTouch	**100**	**62**	**68**	**50**	**85**	**100**	**77.5**	**74.3**

Table 2: Results on four real-world contact-rich manipulation tasks. We report success rates (%). We highlight the **best** and second-best results.

Method	Play Mahjong	Wipe Board	Pick Up Chips	Insert USB	Avg.
ACT	45	15	30	0	22.5
$\pi_{0.5}$	40	45	55	10	37.5
UniVTAC-ACT	70	35	25	0	32.5
FTP-1	95	65	60	**30**	62.5
SimpleTouch	**100**	**85**	**70**	**30**	**71.3**

4.1 Experimental Setup

Benchmarks and Tasks. Our evaluation focuses on contact-rich manipulation in both simulation and the real world. Following FTP-1 (Yuan et al., 2026a), we select six tasks from UniVTAC (Chen et al., 2026a): Lift Bottle, Pull-out Key, Lift Can, Put Bottle, Insert Hole, and Insert Tube. Our real-robot suite comprises Play Mahjong, Wipe Board, Pick Up Chips, and Insert USB; representative rollouts are shown in Figure 3. Every method is trained on the same 50 task-specific demonstrations per task. We evaluate 100 episodes per method and task on UniVTAC and 20 on the real robot.

Baselines and Metrics. On UniVTAC, we compare **SimpleTouch** with seven strong baselines. ACT (Zhao et al., 2023) is a task-trained visuomotor policy and serves as a vision-only reference without policy pretraining, while $\pi_{0.5}$ (Physical Intelligence et al., 2025) provides a strong pretrained VLA reference. VITaL (George et al., 2025) first aligns visual and tactile representations; UniVTAC-ACT (Chen et al., 2026a) augments ACT with tactile observations; and Tactile-VLA (Huang et al., 2025) injects tokenized touch into a pretrained vision-language backbone. FTP-$\pi_{0.5}$ uses the FTP-1 architecture initialized from $\pi_{0.5}$ but without tactile policy pretraining, whereas FTP-1 (Yuan et al., 2026a) is pretrained on large-scale heterogeneous tactile manipulation data. The latter is therefore our principal comparison to a pretrained tactile policy. The real-robot evaluation includes ACT, $\pi_{0.5}$, UniVTAC-ACT, and FTP-1. We report the task success rate and the unweighted average across tasks. Further experimental details are provided in Appendices A, C, and D.

4.2 Main Results

Simulation Results. Table 1 reports results on UniVTAC. With the same 50 task-specific demonstrations, **SimpleTouch** ranks first on all six tasks and reaches an average success rate of 77.5%. This lead spans both visually dominated and contact-sensitive tasks. Among the evaluated tactile policies without tactile policy pretraining, FTP-$\pi_{0.5}$ is the strongest baseline at 45.2%; **SimpleTouch** improves this average by 32.3 percentage points and leads it on every task, with gains ranging from 23 points on Lift Bottle to 42 points on Lift Can. **SimpleTouch** also surpasses the pretrained FTP-1 policy on all six tasks, improving the overall average by 10.8 points. The advantage remains substantial beyond the visually dominated lift tasks: excluding Lift Bottle and Lift Can, **SimpleTouch** averages 74.3%, compared with 59.5% for FTP-1. Its largest gains over FTP-1 occur on Insert Hole and Insert Tube, where it improves success by 21 points on each task.

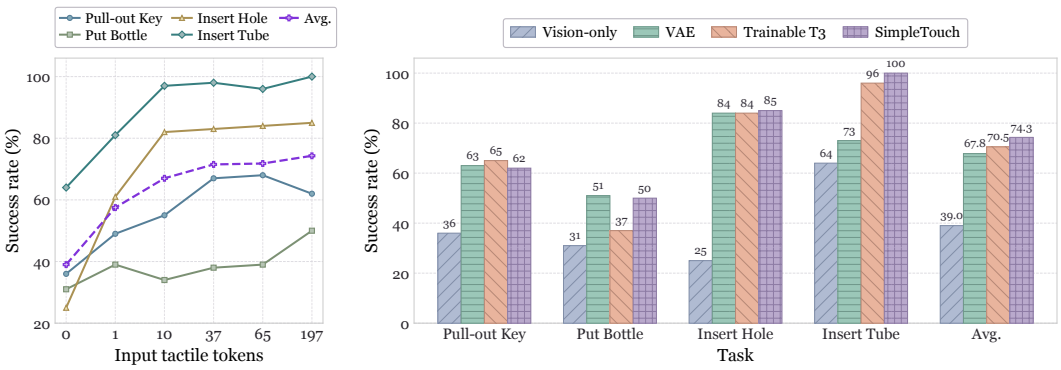

Figure 4: ***(Left)*** Effect of the number of retained current tactile tokens per sensor; fewer tokens imply stronger spatial pooling, 0 is vision-only, 1 retains only the T3 CLS token, and 197 is the full T3 sequence. ***(Right)*** Effects of tactile-encoder variants across tasks; VAE uses a frozen visual VAE, while Trainable T3 fine-tunes T3.

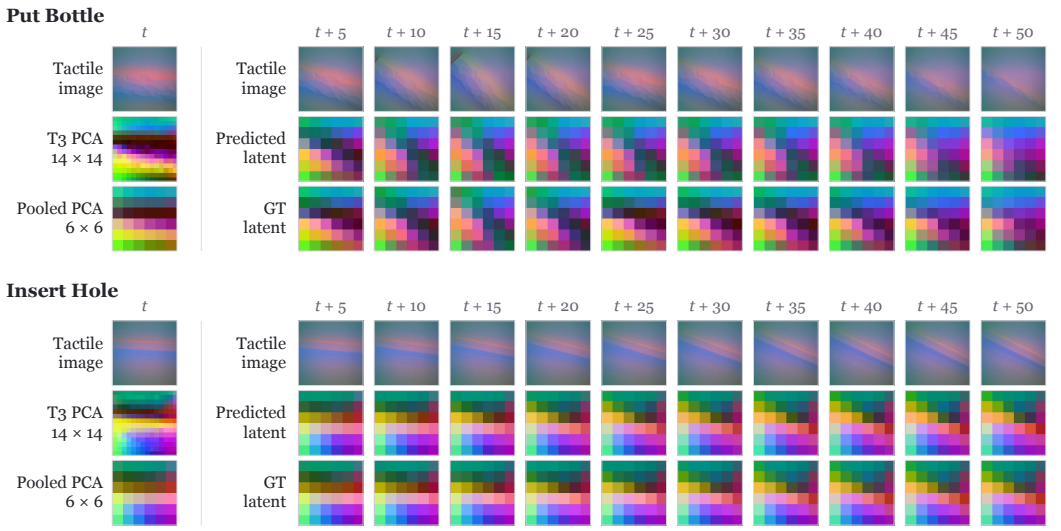

Figure 5: Multi-horizon future tactile prediction. The current column shows the tactile image, the 14×14 spatial T3 PCA map, and its 6×6 pooling. Future columns at 5-step intervals show the observed image and PCA maps of predicted and ground-truth pooled latents.

Real-Robot Results. Across the four real-world tasks in Table 2, **SimpleTouch** raises average success from 62.5% for FTP-1 to 71.3%, an 8.8-percentage-point gain. It leads on Play Mahjong, Wipe Board, and Pick Up Chips, and ties FTP-1 on Insert USB. The 20-point gain on Wipe Board and 10-point gain on Pick Up Chips indicate benefits in sustained-contact and small-object manipulation, respectively. Insert USB remains challenging at 30% for both methods. Together with the UniVTAC results, these experiments show that strong pretrained representations and a task-trained tactile expert can support accurate contact-rich control without tactile policy pretraining.

Qualitative Behavior Analysis. Rollouts reveal a clear progression in how policies use touch across visually ambiguous and dynamically changing contact states. Vision-only policies struggle to resolve ambiguous contact states in the observed rollouts, while policies with weaker tactile integration still misread contact, slip, or apply ineffective corrections. FTP-1 often performs deliberate probing before committing to an action, consistent with contact-acquisition skills encouraged by large-scale tactile policy pretraining; this is particularly valuable for fine-grained manipulation under uncertain contact. **SimpleTouch** shows no equally explicit probing phase, but interprets the available tactile feedback accurately and makes decisive corrections while maintaining stable grasp, contact, and insertion.

4.3 Ablation Studies

Tactile Representation. Figure 4 examines the representation supplied to the tactile expert. Spatial pooling controls the retained token count: fewer tokens reduce spatial resolution, whereas 197 preserves the full T3 sequence. The four-task average rises from 39.0% without touch to 57.5% with only the T3 CLS token and to 74.3% with all 197 tokens. Thus, coarse touch is already useful, but spatial detail brings further gains, especially for Insert Hole, Insert Tube, and Put Bottle. The encoder comparison tests whether the tactile-expert interface remains effective with a different representation family. A pretrained visual VAE reaches 67.8% and remains competitive on three tasks, showing that the interface can exploit a generic visual latent space; its larger drop on Insert Tube (73% versus 100%) shows that demanding contact tasks still benefit from tactile-specialized features. The trainable-T3 variant achieves a mean success rate of 70.5%, below the 74.3% obtained when its pretrained space is preserved, consistent with limited task data partially distorting transferable tactile features.

Future Tactile Prediction. Figure 5 compares observed tactile images with PCA projections of predicted and ground-truth pooled T3 latents. Across both tasks, the predictions qualitatively follow the changing contact structure over multiple future horizons, providing evidence that the tactile expert learns structured tactile evolution rather than an arbitrary auxiliary target. Table 3 compares predictive supervision under the same current-touch representation and action pathway. Multi-horizon prediction raises the four-task average from 68.8% without prediction to 74.3%, a 5.5-point gain, and performs best on every task. Predicting the same pooled representation only at $t{+}50$ reaches 68.5%, compared with 74.3% for ten future horizons. With spatial resolution held fixed, this 5.8-point gain supports supervising intermediate contact evolution rather than only the final future state. The largest gain over no prediction is on Put Bottle (50% versus 37%), where contact changes continuously during placement; Insert Hole and Insert Tube show smaller but consistent gains after contact. Together, the qualitative fidelity and policy improvements support multi-horizon prediction as supervision for the tactile-action interface.

Table 3: Future tactile prediction ablations on UniVTAC. We highlight the **best** and second-best values.

Variant	Pull-out Key	Put Bottle	Insert Hole	Insert Tube	Avg.
Vision-only	36	31	25	64	39.0
w/o Tac. Pred.	60	37	83	95	68.8
Single-horizon	53	44	79	98	68.5
SimpleTouch	**62**	**50**	**85**	**100**	**74.3**

4.4 Data Efficiency

To test whether **SimpleTouch** remains effective when task data are scarce, we compare it with FTP-$\pi_{0.5}$ on Put Bottle and Insert Hole across four demonstration budgets. Figure 6 shows a consistent advantage at every budget. With only 10 demonstrations, **SimpleTouch** reaches 23% and 43%, versus 5% and 14% for FTP-$\pi_{0.5}$. With 20 demonstrations—60% fewer than the baseline's 50-demonstration setting—it reaches 38% and 57%, exceeding the corresponding baseline results of 19% and 47%. The advantage therefore emerges well before the standard 50-demonstration setting. Performance is not universally monotonic at larger budgets, so these results support consistent data-efficiency gains rather than a general scaling law.

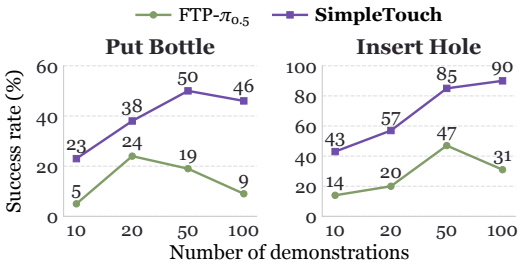

Figure 6: Data efficiency on UniVTAC. Success rates of FTP-$\pi_{0.5}$ and **SimpleTouch** on Put Bottle and Insert Hole across demonstration budgets.

Table 4: Tactile design choices in FTP-$\pi_{0.5}$ and **SimpleTouch**; shaded rows mark design choices that differ between the two methods.

Design choice	FTP-$\pi_{0.5}$	SimpleTouch
Tactile encoder	T3	T3
Encoder update	Fine-tuned	Frozen
Current tactile tokens	CLS only	All tokens
Tactile expert	Yes	Yes
Future tactile prediction	No	Multi-horizon

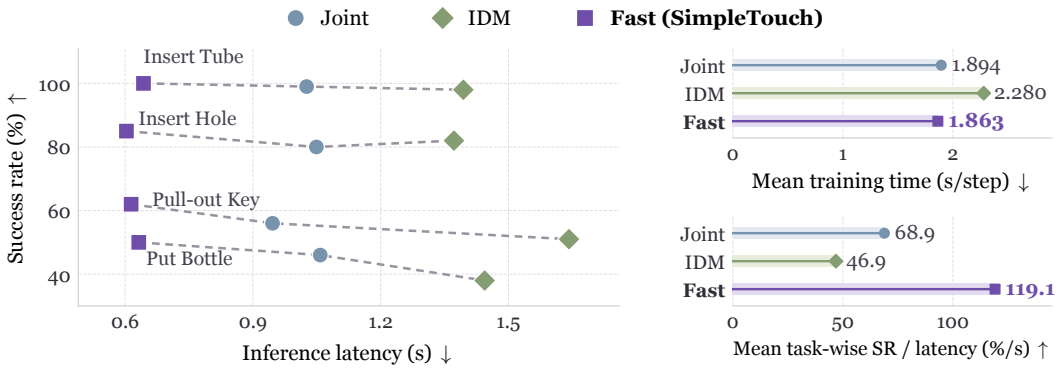

Figure 7: Training and inference efficiency of Joint, IDM, and Fast. The left panel shows success rate versus inference latency; dashed lines connect formulations evaluated on the same task. The right panels show mean training time (top) and mean task-wise success-rate/latency ratio (bottom).

Table 4 helps interpret this gap. Both methods use T3 and a tactile expert, focusing the comparison on how tactile representations are preserved and supervised rather than whether touch is present. The ablations in Section 4.3 support each distinction: retaining all tactile tokens preserves spatial contact detail, keeping the pretrained encoder fixed performs better than updating it in this limited-data setting, and multi-horizon prediction provides dense supervision for learning contact dynamics. Together, these choices help explain why **SimpleTouch** remains effective across demonstration budgets, including the lowest-data settings.

4.5 Training and Inference Efficiency

We evaluate three formulations adapted from Fast-WAM (Yuan et al., 2026b). All use future tactile prediction as training supervision but differ in how future latents participate in action generation: *Joint* denoises tactile latents and actions together, *IDM* generates future latents before predicting actions, and *Fast*, used by **SimpleTouch**, predicts actions directly from cached current context. Figure 7 summarizes their success rates and timings on the four ablation tasks.

Fast establishes the strongest accuracy–efficiency trade-off. It attains lower latency and higher success on every task, rather than improving the average through one favorable case. Its mean success rate is 74.3%, versus 70.3% for Joint and 67.3% for IDM, while mean latency falls to 0.623 s from 1.020 s and 1.463 s—reductions of 38.9% and 57.4%. This consistent upper-left shift suggests that feeding generated future latents into the action pathway is unnecessary and may introduce an avoidable dependency on prediction quality. Future-touch prediction is more effective here as a training signal for the tactile expert, while deployment relies on observed tactile context.

The conclusion holds when training cost is considered. Fast and Joint require 1.863 s and 1.894 s per measured training operation, a difference of only 1.6%, whereas IDM requires 2.280 s. Combining task success with latency, Fast reaches 119.1 %/s, compared with 68.9 for Joint and 46.9 for IDM—1.73 and 2.54 times as high, respectively. Fast therefore preserves predictive supervision without paying for future tactile generation at deployment, motivating its use as the default formulation in **SimpleTouch**. Timing definitions, hardware, and protocols are provided in Appendix E.3.

5 Conclusion

We present **SimpleTouch**, a simple and effective recipe for tactile-enhanced vision-language-action policies without additional tactile policy pretraining or separate alignment. It augments $\pi_{0.5}$ with a tactile expert that consumes the full token sequence of a pretrained tactile encoder, learns through action supervision and multi-horizon future-touch prediction, and acts from current tactile context at inference. With 50 demonstrations per task, **SimpleTouch** leads all six UniVTAC tasks (77.5% average) and reaches 71.3% across four real-robot tasks, outperforming tactile-policy-pretrained FTP-1 by 10.8 and 8.8 percentage points, respectively. Ablations support preserved spatial detail, stable pretrained representations, and predictive supervision as useful ingredients, providing a practical and efficient route to contact-rich manipulation.

AI USE STATEMENT

Generative AI tools were used to assist with literature retrieval, research ideation and experimental planning, software implementation, result analysis and visualization, and the drafting and polishing of this manuscript. All AI-assisted code, citations, analyses, figures, and text were reviewed and verified by the authors. The authors take full responsibility for the content and claims of this work.

ETHICS STATEMENT

This work studies tactile-enhanced robot manipulation in simulation and on real robotic platforms. The experiments involve no human subjects, personal data, or deployment in safety-critical environments, and all real-robot evaluations were conducted under controlled laboratory conditions. We do not identify direct ethical risks beyond the standard safety considerations associated with robotic manipulation.

REPRODUCIBILITY STATEMENT

We support reproducibility by describing the model and learning objectives in Section 3, the evaluation protocol and principal experimental settings in Section 4, and additional implementation, training, hardware, and evaluation details in the appendices. We are preparing a public release of the complete codebase and model checkpoints, together with all training configurations and evaluation scripts, and will make these resources available in the near future.

REFERENCES

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, et al. PaliGemma: A Versatile 3B VLM for Transfer, 2024. URL https://arxiv.org/abs/2407.07726.

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, Hongyan Zhao, Hanyu Liu, Zhizhong Su, Lei Ma, Hang Su, and Jun Zhu. Motus: A Unified Latent Action World Model, 2025a. URL https://arxiv.org/abs/2512.13030.

Jianxin Bi, Kevin Yuchen Ma, Ce Hao, Mike Zheng Shou, and Harold Soh. VLA-Touch: Enhancing Vision-Language-Action Models with Dual-Level Tactile Feedback, 2025b. URL https://arxiv.org/abs/2507.17294.

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, 2024. URL https://arxiv.org/abs/2410.24164.

Rui Cai, Jun Guo, Xinze He, Piaopiao Jin, Jie Li, Bingxuan Lin, Futeng Liu, Wei Liu, Fei Ma, Kun Ma, Feng Qiu, Heng Qu, Yifei Su, Qiao Sun, Dong Wang, Donghao Wang, Yunhong Wang, Rujie Wu, Diyun Xiang, Yu Yang, Hangjun Ye, Yuan Zhang, and Quanyun Zhou. Xiaomi-Robotics-0: An Open-Sourced Vision-Language-Action Model with Real-Time Execution, 2026. URL https://arxiv.org/abs/2602.12684.

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, 2026a. URL https://arxiv.org/abs/2602.10093.

Jialei Chen, Kai Wang, Kang Chen, Shuaihang Chen, Feng Gao, Wenhao Tang, Zhiyuan Li, Weilin Liu, Zhuyu Yao, Boxun Li, Yuanbo Xu, and Chao Yu. LaWAM: Latent World Action Models for

Efficient Dynamics-Aware Robot Policies, 2026b. URL https://arxiv.org/abs/2606.15768.

Ruilin Chen, Jingkai Jia, Tong Yang, Xinyu Zhou, Qiao Sun, Jiangwei Zhong, Shizeng Zhang, Nuo Chen, Bailin He, Wei Li, and Wenqiang Zhang. Representation-Aligned Tactile Grounding for Contact-Rich Robotic Manipulation, 2026c. URL https://arxiv.org/abs/2607.14609.

Xinyi Chen, Yilun Chen, Yanwei Fu, Ning Gao, Jiaya Jia, Weiyang Jin, Hao Li, Yao Mu, Jiangmiao Pang, Yu Qiao, Yang Tian, Bin Wang, Bolun Wang, Fangjing Wang, Hanqing Wang, Tai Wang, Ziqin Wang, Xueyuan Wei, Chao Wu, Shuai Yang, Jinhui Ye, Junqiu Yu, Jia Zeng, Jingjing Zhang, Jinyu Zhang, Shi Zhang, Feng Zheng, Bowen Zhou, and Yangkun Zhu. InternVLA-M1: A Spatially Guided Vision-Language-Action Framework for Generalist Robot Policy, 2025. URL https://arxiv.org/abs/2510.13778.

Ning Cheng, Jinan Xu, Changhao Guan, Jing Gao, Weihao Wang, You Li, Fandong Meng, Jie Zhou, Bin Fang, and Wenjuan Han. Touch100K: A Large-Scale Touch-Language-Vision Dataset for Touch-Centric Multimodal Representation. *Information Fusion*, 124:103305, 2025a. doi: 10.1016/j.inffus.2025.103305. URL https://doi.org/10.1016/j.inffus.2025.103305.

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, 2025b. URL https://arxiv.org/abs/2508.08706.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems XIX*, 2023. doi: 10.15607/RSS.2023.XIX.026. URL https://diffusion-policy.cs.columbia.edu/diffusion_policy_2023.pdf.

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, 2025. URL https://arxiv.org/abs/2502.12191.

Ruoxuan Feng, Yuxuan Zhou, Siyu Mei, Dongzhan Zhou, Pengwei Wang, Shaowei Cui, Bin Fang, Guocai Yao, and Di Hu. AnyTouch 2: General Optical Tactile Representation Learning For Dynamic Tactile Perception, 2026. URL https://arxiv.org/abs/2602.09617.

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. doi: 10.1109/ICRA55743.2025.11128336. URL https://arxiv.org/abs/2403.11898.

Irmak Guzey, Ben Evans, Soumith Chintala, and Lerrel Pinto. Dexterity from Touch: Self-Supervised Pre-Training of Tactile Representations with Robotic Play. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 3142–3166. PMLR, 2023. URL https://proceedings.mlr.press/v229/guzey23a.html.

Liang Heng, Haoran Geng, Kaifeng Zhang, Pieter Abbeel, and Jitendra Malik. ViTacFormer: Learning Cross-Modal Representation for Visuo-Tactile Dexterous Manipulation, 2025. URL https://arxiv.org/abs/2506.15953.

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, 2024. URL https://arxiv.org/abs/2410.24090.

Carolina Higuera, Akash Sharma, Taosha Fan, Chaithanya Krishna Bodduluri, Byron Boots, Michael Kaess, Mike Lambeta, Tingfan Wu, Zixi Liu, Francois Robert Hogan, and Mustafa Mukadam. Tactile Beyond Pixels: Multisensory Touch Representations for Robot Manipulation. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 105–123. PMLR, 2025. URL https://proceedings.mlr.press/v305/higuera25a.html.

Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video Prediction Policy: A Generalist Robot Policy with Predictive Visual Representations, 2024. URL https://arxiv.org/abs/2412.14803.

Binghao Huang, Yixuan Wang, Xinyi Yang, Yiyue Luo, and Yunzhu Li. 3D-ViTac: Learning Fine-Grained Manipulation with Visuo-Tactile Sensing, 2024. URL https://arxiv.org/abs/2410.24091.

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, 2025. URL https://arxiv.org/abs/2507.09160.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, et al. DROID: A Large-Scale In-The-Wild Robot Manipulation Dataset, 2024. URL https://arxiv.org/abs/2403.12945.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success, 2025a. URL https://arxiv.org/abs/2502.19645.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025b. URL https://proceedings.mlr.press/v270/kim25c.html.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, Yujun Shen, and Yinghao Xu. Causal World Modeling for Robot Control, 2026a. URL https://arxiv.org/abs/2601.21998.

Rui Li, Robert Platt, Wenzhen Yuan, Andreas ten Pas, Nathan Roscup, Mandayam A. Srinivasan, and Edward Adelson. Localization and manipulation of small parts using GelSight tactile sensing. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3988–3993. IEEE, 2014. doi: 10.1109/IROS.2014.6943123. URL https://doi.org/10.1109/IROS.2014.6943123.

Ruicheng Li, Qixiu Li, Ruichun Ma, Yu Deng, Lin Luo, Zhiying Du, Jianfeng Xiang, Huizhi Liang, Ruicheng Wang, Jiaolong Yang, and Baining Guo. FM-VLA: Force-based Memory for Vision-Language-Action Models in Contact-Rich Manipulation, 2026b. URL https://arxiv.org/abs/2607.18231.

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, 2026c. URL https://arxiv.org/abs/2605.07308.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Fangchen Liu, Chuanyu Li, Yihua Qin, Jing Xu, Pieter Abbeel, and Rui Chen. ViTaMIn: Learning Contact-Rich Tasks Through Robot-Free Visuo-Tactile Manipulation Interface, 2025a. URL https://arxiv.org/abs/2504.06156.

Qingtao Liu, Qi Ye, Zhengnan Sun, Yu Cui, Gaofeng Li, and Jiming Chen. Masked Visual-Tactile Pre-training for Robot Manipulation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13859–13875, 2024. doi: 10.1109/ICRA57147.2024.10610933.

Qingtao Liu, Yu Cui, Zhengnan Sun, Gaofeng Li, Jiming Chen, and Qi Ye. VTDexManip: A Dataset and Benchmark for Visual-Tactile Pretraining and Dexterous Manipulation with Reinforcement Learning. In *International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=jf7C7EGw21.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1711.05101.

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, 2026. URL https://arxiv.org/abs/2606.08737.

Charlotte Morissette, Amin Abyaneh, Wei-Di Chang, Anas Houssaini, David Meger, Hsiu-Chin Lin, Jonathan Tremblay, and Gregory Dudek. Tactile Modality Fusion for Vision-Language-Action Models, 2026. URL https://arxiv.org/abs/2603.14604.

NeoteAI Team and Fudan TEAI Team. N_0-TWAM: Scaling Tactile-Native World-Action Model for Contact-Rich Manipulation, 2026a. URL https://arxiv.org/abs/2607.23783.

NeoteAI Team and Fudan TEAI Team. N_0-VTLA: Scaling Vision-Tactile-Language-Action Model with Latent Tactile Tokens, 2026b. URL https://arxiv.org/abs/2607.23782.

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, Fei-Fei Li, Ken Goldberg, Jitendra Malik, Pieter Abbeel, Yuke Zhu, Danfei Xu, Linxi Fan, and Trevor Darrell. T-Rex: Tactile-Reactive Dexterous Manipulation, 2026a. URL https://arxiv.org/abs/2606.17055.

Yaru Niu, Zhenlong Fang, Binghong Chen, Shuai Zhou, Revanth Krishna Senthilkumaran, Hao Zhang, Bingqing Chen, Chen Qiu, H. Eric Tseng, Jonathan Francis, and Ding Zhao. Learning Versatile Humanoid Manipulation with Touch Dreaming, 2026b. URL https://arxiv.org/abs/2604.13015.

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, et al. GR00T N1: An Open Foundation Model for Generalist Humanoid Robots, 2025. URL https://arxiv.org/abs/2503.14734.

Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, et al. Open X-Embodiment: Robotic Learning Datasets and RT-X Models, 2023. URL https://arxiv.org/abs/2310.08864.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: Efficient Action Tokenization for Vision-Language-Action Models, 2025. URL https://arxiv.org/abs/2501.09747.

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, 2025. URL https://arxiv.org/abs/2504.16054.

StarVLA Community. StarVLA: A Lego-like Codebase for Vision-Language-Action Model Developing, 2026. URL https://arxiv.org/abs/2604.05014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, 2017. URL https://arxiv.org/abs/1706.03762.

Longyan Wu, Checheng Yu, Jieji Ren, Li Chen, Yufei Jiang, Ran Huang, Guoying Gu, and Hongyang Li. FreeTacMan: Robot-Free Visuo-Tactile Data Collection System for Contact-Rich Manipulation, 2025. URL https://arxiv.org/abs/2506.01941.

Congsheng Xu, Qiaochu Yang, Fangyuan Shi, Yifan Han, Baijun Chen, Yiming Wang, Haonan Zhao, Zhe Liu, Yao Mu, Daolin Ma, Xiaokang Yang, and Hesheng Wang. VT-MUSE: Multimodal Unified Sequential Visuotactile Representation Learning for Manipulation, 2026. URL https://arxiv.org/abs/2608.21290.

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, 2025. URL https://arxiv.org/abs/2503.02881.

Fengyu Yang, Chao Feng, Ziyang Chen, Hyoungseob Park, Daniel Wang, Yiming Dou, Ziyao Zeng, Xien Chen, Rit Gangopadhyay, Andrew Owens, and Alex Wong. Binding Touch to Everything: Learning Unified Multimodal Tactile Representations, 2024. URL https://arxiv.org/abs/2401.18084.

Guo Ye, Zexi Zhang, Xu Zhao, Shang Wu, Haoran Lu, Shihan Lu, and Han Liu. Learning to Feel the Future: DreamTacVLA for Contact-Rich Manipulation, 2025. URL https://arxiv.org/abs/2512.23864.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, Ayaan Malik, Kyungmin Lee, William Liang, Nadun Ranawaka, Jiasheng Gu, Yinzhen Xu, Guanzhi Wang, Fengyuan Hu, Avnish Narayan, Johan Bjorck, Jing Wang, Gwanghyun Kim, Dantong Niu, Ruijie Zheng, Yuqi Xie, Jimmy Wu, Qi Wang, Ryan Julian, Danfei Xu, Yilun Du, Yevgen Chebotar, Scott Reed, Jan Kautz, Yuke Zhu, Linxi "Jim" Fan, and Joel Jang. World Action Models are Zero-shot Policies, 2026. URL https://arxiv.org/abs/2602.15922.

Jiawen Yu, Hairuo Liu, Qiaojun Yu, Jieji Ren, Ce Hao, Haitong Ding, Guangyu Huang, Guofan Huang, Yan Song, Panpan Cai, Cewu Lu, and Wenqiang Zhang. ForceVLA: Enhancing VLA Models with a Force-aware MoE for Contact-rich Manipulation. In *Advances in Neural Information Processing Systems*, 2025. doi: 10.48550/ARXIV.2505.22159. URL https://arxiv.org/abs/2505.22159.

Kelin Yu, Yunhai Han, Qixian Wang, Vaibhav Saxena, Danfei Xu, and Ye Zhao. MimicTouch: Leveraging Multi-modal Human Tactile Demonstrations for Contact-rich Manipulation, 2023. URL https://arxiv.org/abs/2310.16917.

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, 2026a. URL https://arxiv.org/abs/2606.13102.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-WAM: Do World Action Models Need Test-time Future Imagination?, 2026b. URL https://arxiv.org/abs/2603.16666.

Wenzhen Yuan, Siyuan Dong, and Edward Adelson. GelSight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12):2762, 2017. doi: 10.3390/s17122762. URL https://doi.org/10.3390/s17122762.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid Loss for Language Image Pre-Training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. doi: 10.1109/ICCV51070.2023.01100. URL https://arxiv.org/abs/2303.15343.

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, 2026a. URL https://arxiv.org/abs/2603.12665.

Qihang Zhang, Lin Li, Luyao Zhang, Shuai Yang, Yiming Luo, Shuaiting Li, Ruilin Wang, Junke Wang, Jiahao Shao, Gangwei Xu, Jiaming Zhou, Yishu Shen, Yudong Jin, Fangyi Xu, Shuailei

Ma, Jiaqi Liao, Guanxing Lu, Zifan Shi, Yongkun Wen, Yujie Zhao, Weixuan Tang, Xinyang Wang, Chaojian Li, Jiapeng Zhu, Ka Leong Cheng, Nan Xue, Xing Zhu, Yujun Shen, and Yinghao Xu. Native Video-Action Pretraining for Generalizable Robot Control, 2026b. URL https://arxiv.org/abs/2607.08639.

Xidong Zhang, Yichi Zhang, Jiaxin Shi, Fucai Zhu, Siyu Zhu, Michael Yu Wang, Xiaojun Wu, and Weihao Yuan. UniTacVLA: Unified Tactile Understanding and Prediction in Vision Language Action Models, 2026c. URL https://arxiv.org/abs/2606.31723.

Jialiang Zhao, Yuxiang Ma, Lirui Wang, and Edward Adelson. Transferable Tactile Transformers for Representation Learning Across Diverse Sensors and Tasks. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3766–3779. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/zhao25c.html.

Tony Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Robotics: Science and Systems XIX*, RSS2023. Robotics: Science and Systems Foundation, July 2023. doi: 10.15607/rss.2023.xix.016. URL http://dx.doi.org/10.15607/RSS.2023.XIX.016.

Yuhang Zheng, Songen Gu, Yupeng Zheng, Weize Li, Yujie Zang, Shuai Tian, Xiang Li, Ce Hao, Chen Gao, Si Liu, Haoran Li, Yilun Chen, Shuicheng Yan, and Wenchao Ding. OmniVTA: Visuo-Tactile World Modeling for Contact-Rich Robotic Manipulation, 2026. URL https://arxiv.org/abs/2603.19201.

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, 2026. URL https://arxiv.org/abs/2607.07287.

A Implementation Details

A.1 Model, Training, and Inference Configuration

Unless otherwise specified, all **SimpleTouch** experiments use the model, training, and inference configuration summarized in Table 5. The values report the resolved default configuration used by the training launcher rather than inactive model defaults.

Table 5: Model, training, and inference configurations for **SimpleTouch**.

Configuration	Value
Vision-language-action backbone	
Base policy	$\pi_{0.5}$
Visual-language backbone	PaliGemma (Beyer et al., 2024): SigLIP (Zhai et al., 2023) vision encoder and Gemma 2B language model
Action expert	Gemma 300M; width 1024; 18 Transformer layers (Vaswani et al., 2017)
State and action spaces	Shared 8-dimensional interface: UniVTAC uses seven arm joints and one gripper coordinate; PiPER uses six arm joints, one zero dummy joint, and one gripper coordinate. States are absolute; action-joint dimensions are relative and the gripper action is absolute. Both are zero-padded to 32 internal model dimensions
Action chunk length	50
Tactile representation and expert	
Tactile encoder	Pretrained T3-large; frozen during task training
Tactile preprocessing	Raw RGB; resize, center crop to 224×224, and task-specific normalization
Current tactile tokens	2 sensors $\times$ (1 CLS + 14×14 spatial tokens) = 394 tokens; feature dimension 1024
Tactile expert	Independently initialized; same width and depth as the action expert
Prediction formulation	Fast; actions attend to current tactile features but not future tactile targets
Future horizons	$+5, +10, \ldots, +50$
Future target tokens	2 sensors $\times$ (1 CLS + 6×6 pooled spatial tokens) per horizon; 740 tokens in total
Auxiliary objective	Flow-matching velocity MSE; $\lambda_T = 0.01$ with a constant schedule
Optimization scope	Action and tactile objectives jointly update the VLA and tactile expert
Episode-tail supervision	Repeat the final valid action and tactile target and retain supervision
Training configuration	
Task data	50 task-specific demonstrations per task
Training duration	20,000 updates; checkpoint every 5,000 updates
Optimizer	AdamW (Loshchilov & Hutter, 2019); $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$
Learning rate	Peak 5×10^{-5}; 1,000-step warmup; cosine decay to 5×10^{-6}
Weight decay / gradient clipping	10^{-10} / global norm 1.0
Batch size	8 samples per GPU $\times$ 8 GPUs = 64
Training hardware	8 NVIDIA H20 GPUs
Inference configuration	
Flow solver	10 Euler integration steps
Prediction / execution horizon	Predict 50 actions; execute 25 for Insert Hole and Insert Tube and 50 for the other tasks
Tactile computation	Encode current touch once per policy query and reuse its layerwise KV cache during action denoising
Additional inference operations	No future tactile generation or temporal ensembling

A.2 Tactile Expert Training and Inference

This section gives the exact token construction, conditioning graph, and optimization objective used by the Fast formulation of **SimpleTouch**. It also makes explicit which computations are retained at deployment. These details complement the architectural description in Section 3; they do not introduce an additional training stage.

Current and Future Tactile Tokens. Let E_ϕ denote the pretrained T3 encoder and let U_t^l and U_t^r be the synchronized observations from the two tactile sensors. For current touch, we concatenate the complete output of both sensors,

$$Z_t = \left[E_\phi(U_t^l); E_\phi(U_t^r)\right] \in \mathbb{R}^{394\times 1024}. \tag{8}$$

Each sensor contributes one global token and a 14×14 spatial grid. No pooling is applied to Z_t, and a learned sensor-identity embedding distinguishes the two streams. Future prediction uses the offsets $\Delta = \{5, 10, \ldots, 50\}$. For each sensor and offset, the global token is retained and the 14×14 spatial grid is pooled to 6×6. The resulting target is

$$Y_t = \operatorname*{Stack}_{\delta\in\Delta} \operatorname*{Stack}_{s\in\{l,r\}} \mathcal{S}\big(\mathcal{P}(E_\phi(U_{t+\delta}^s))\big) \in \mathbb{R}^{10\times 2\times 37\times 1024}, \tag{9}$$

where $\mathcal{P}$ denotes spatial pooling and $\mathcal{S}(x) = x/r$ uses the checkpoint-derived reference

$$r = \sqrt{\frac{1}{D}\sum_{d=1}^{D}(\gamma_d^2 + \beta_d^2)}, \tag{10}$$

with $\gamma, \beta \in \mathbb{R}^D$ from the final LayerNorm of the pretrained T3 encoder. This fixed scaling is applied only to future targets; the current tactile context in Eq. (8) remains unmodified. The horizon and sensor axes are flattened into 740 future tokens before the tactile expert. Near the end of a trajectory, the final valid action and tactile observation are repeated so that all horizons retain supervision.

Independent Flow Paths. We train the action chunk A_t and future tactile target Y_t with two conditional flow-matching paths. For $q \in \{A, T\}$, let x_q denote the corresponding clean target, $\epsilon_q \sim \mathcal{N}(0, I)$, and $\tau_q \sim \mathrm{Beta}(1.5, 1)$. The action and tactile times are sampled independently and mapped to $[0.001, 1]$. Their interpolants and velocity targets are

$$x_q^{\tau_q} = \tau_q\epsilon_q + (1-\tau_q)x_q, \qquad v_q^\star = \epsilon_q - x_q. \tag{11}$$

The two losses are

$$\mathcal{L}_{\mathrm{act}} = \mathbb{E}[\mathrm{MSE}(f_A(x_A^{\tau_A}, \tau_A; C_t, Z_t), v_A^\star)], \tag{12}$$

$$\mathcal{L}_{\mathrm{tac}} = \mathbb{E}[\mathrm{MSE}(f_T(x_T^{\tau_T}, \tau_T; C_t, Z_t), v_T^\star)], \tag{13}$$

where C_t is the visual-language and state context and MSE averages the squared error over the complete velocity tensor. Thus, the tactile expert reads the full noised future sequence, including all horizons, while every equally sized horizon receives equal weight. The final objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{act}} + \lambda_T\mathcal{L}_{\mathrm{tac}}, \qquad \lambda_T = 0.01. \tag{14}$$

The action and tactile branches share the same visual-language context and current-touch cache, but not their noise or flow time. Consequently, tactile prediction supplies an additional learning signal without forcing action denoising to follow the prediction trajectory.

Block Attention Structure. The Fast formulation uses a fixed asymmetric visibility pattern. Rows below denote query blocks and columns denote the key-value blocks they may read:

query\key/value	C_t	Z_t	$\widetilde{Y}_t$	$\widetilde{A}_t$
C_t	✓	–	–	–
Z_t	–	✓	–	–
$\widetilde{Y}_t$	✓	✓	✓	–
$\widetilde{A}_t$	✓	✓	–	✓

(15)

Here $\widetilde{Y}_t$ and $\widetilde{A}_t$ are the noised future-tactile and action tokens. The four blocks correspond to c_0, z_0, $z_{1:H}$, and $a_{1:H}$ in Figure 2; its future-tactile labels schematically span the sampled offsets Δ, not every action step. Current tactile queries are deliberately isolated from the visual-language stream, preserving a modality-specific tactile computation. Future-tactile queries can use the current multimodal context to model contact evolution. Action queries read current touch but never the future-tactile branch; conversely, future-tactile queries never read action tokens. Thus, future tactile observations act as prediction targets rather than privileged action inputs.

Fast Inference and Cache Reuse. At each policy query, the visual-language backbone first produces the layerwise context cache $\mathcal{K}_C$. The tactile expert processes Z_t once using tactile self-attention with the fixed clean-time condition $\tau = 0$ and produces $\mathcal{K}_T = \mathcal{K}_t$ from Eq. (3); we suppress the environment-step index t on cache components below. At every Transformer layer, their keys and values are separately concatenated along the sequence dimension,

$$\mathcal{K}^j_{C,T} = \left(\mathrm{Concat}_{\mathrm{seq}}(K^j_C, K^j_T), \mathrm{Concat}_{\mathrm{seq}}(V^j_C, V^j_T)\right), \qquad j = 1, \ldots, L. \tag{16}$$

The action solver samples $x^0_A \sim \mathcal{N}(0, I)$ and performs $N = 10$ Euler steps,

$$x^{k+1}_A = x^k_A - \frac{1}{N} f_A\left(x^k_A, 1 - \frac{k}{N}; \mathcal{K}_{C,T}\right), \qquad k = 0, \ldots, N-1. \tag{17}$$

The clean context cache is reused at every step because neither C_t nor Z_t depends on action noise or flow time. The future-tactile branch, its noisy tokens, and its output projection are omitted at inference. Multi-horizon prediction therefore trains the same tactile expert that constructs $\mathcal{K}_T$, while deployment retains only current-touch encoding and action generation.

B Extended Related Work

This section expands the three research directions summarized in Section 2, retaining the same organization and distinction between policy learning and tactile representation learning.

B.1 Generalist Policy Learning

Task-specific imitation learning established strong sequence-modeling and generative-control baselines through action chunking and diffusion (Zhao et al., 2023; Chi et al., 2023). Generalist policy learning broadens this setting by combining heterogeneous robot data with pretrained vision-language representations. Open X-Embodiment and DROID provide large robot-interaction corpora (Open X-Embodiment Collaboration et al., 2023; Khazatsky et al., 2024), while OpenVLA, the π family, GR00T N1, and related systems develop reusable policies across tasks and embodiments (Kim et al., 2025b; Black et al., 2024; Physical Intelligence et al., 2025; NVIDIA et al., 2025). FAST action tokenization and OFT further study efficient action representations and task fine-tuning (Pertsch et al., 2025; Kim et al., 2025a). These systems primarily organize vision, language, proprioception, and actions; local tactile contact is generally not native to their pretrained observation interfaces.

A complementary world-action-model line incorporates predictive visual or latent dynamics into control. Video Prediction Policy predicts future visual features (Hu et al., 2024), whereas Motus, DreamZero, LingBot-VA, LaWAM, and Fast-WAM couple visual or latent prediction with action generation in different training and inference formulations (Bi et al., 2025a; Ye et al., 2026; Li et al., 2026a; Zhang et al., 2026b; Chen et al., 2026b; Yuan et al., 2026b). Collectively, these works motivate prediction as supervision beyond behavioral cloning, but they still predominantly model visual rather than tactile dynamics.

B.2 Tactile Policy Learning

As in Section 2, we organize tactile policies by where their contact-sensitive priors are acquired. *Tactile policy pretraining and mid-training* learn tactile control before downstream adaptation: FTP-1 and the N_0 family scale tactile-native policy learning, T-Rex introduces tactile-grounded

mid-training and fast reaction, and TouchWorld combines tactile prediction with hierarchical control (Yuan et al., 2026a; NeoteAI Team & Fudan TEAI Team, 2026a;b; Niu et al., 2026a; Zhou et al., 2026). *Representation alignment* instead reduces the modality gap before policy learning through shared, sequential, or semantically aligned visual-tactile spaces, as explored by VITaL, UniTouch, VT-MUSE, and OmniVTLA (George et al., 2025; Yang et al., 2024; Xu et al., 2026; Cheng et al., 2025b).

Task-level tactile integration directly equips pretrained policies with contact observations. Tactile-VLA, AT-VLA, and TacFiLM respectively emphasize tactile reasoning, adaptive injection, and feature-wise modulation, while ForceVLA and FM-VLA use force-aware expert routing or force-history memory (Huang et al., 2025; Li et al., 2026c; Morissette et al., 2026; Yu et al., 2025; Li et al., 2026b). Other systems similarly study dual-level feedback, structured visuotactile features, or fast tactile reaction (Bi et al., 2025b; Huang et al., 2024; Heng et al., 2025; Xue et al., 2025).

Predictive tactile learning models future contact as supervision or a control signal. Dream-Tac, DreamTacVLA, Representation-Aligned Tactile Grounding, UniTacVLA, Touch Dreaming, and OmniVTA span visuotactile world models, future-touch prediction, representation-level grounding, and tactile-conditioned correction (Lou et al., 2026; Ye et al., 2025; Chen et al., 2026c; Zhang et al., 2026c; Niu et al., 2026b; Zheng et al., 2026). Although these categories can overlap, they distinguish whether additional tactile knowledge comes from policy pretraining, representation alignment, task-level fusion, or predictive supervision.

B.3 Pretrained Tactile Encoders

Pretrained tactile encoders address sensory representation acquisition rather than action generation. T-Dex learns self-supervised embeddings from robotic play (Guzey et al., 2023); Sparsh and Sparsh-X study self-supervised visual-tactile and multisensory representations (Higuera et al., 2024; 2025); and UniTouch and the AnyTouch series target transfer across sensors, modalities, and temporal tactile signals (Yang et al., 2024; Feng et al., 2025; 2026). T3 combines sensor-specific input processing with a shared Transformer trained across diverse sensors and tasks (Zhao et al., 2025). Masked visual-tactile modeling, tactile dexterity benchmarks, and large-scale touch-centric datasets provide complementary objectives and resources for reusable tactile features (Liu et al., 2024; 2025b; Cheng et al., 2025a).

Data collection systems and simulators also shape the resulting representations. Robot-free interfaces such as ViTaMIn and FreeTacMan expand tactile interaction data without conventional robot teleoperation (Liu et al., 2025a; Wu et al., 2025), while UniVTAC supplies controlled visuotactile trajectories and standardized evaluation (Chen et al., 2026a). Such encoders provide transferable contact features, but do not prescribe how those features should influence a pretrained action generator. This separates tactile representation pretraining from tactile policy pretraining and motivates studying the downstream policy interface independently.

C Benchmarks and Evaluation Protocols

C.1 UniVTAC Tasks and Demonstrations

The main simulation study follows the six-task UniVTAC subset used by FTP-1 (Chen et al., 2026a; Yuan et al., 2026a). All evaluated methods use the same 50 official task-specific demonstrations and are evaluated for 100 episodes per task. Each task has a separately fine-tuned policy, and no trajectories are shared across tasks.

Action Prediction and Execution Horizons. Table 6 separates the 50-step action chunk predicted at each policy query from the number of actions executed before requesting a new observation. We specify the execution horizon before evaluation from task characteristics rather than tuning it on validation or test performance. The non-insertion tasks execute the full predicted chunk, whereas Insert Hole and Insert Tube execute only the first 25 actions. These insertion tasks require repeated contact-conditioned corrections while the object is aligned with a constrained opening. The shorter execution horizon refreshes the tactile observation midway through the predicted chunk, allowing the policy to reassess both the alignment error and the appropriate correction direction after contact

changes. Executing a longer open-loop prefix can instead preserve a stale correction after the contact geometry has changed, leading to overshoot, jamming, slippage, or the object moving away from the insertion region. All tasks retain the same prediction horizon and ten action-flow integration steps; only the feedback frequency differs. Baseline results use the official default execution settings of their respective methods.

Table 6: UniVTAC task protocol for **SimpleTouch**. "Predicted" and "Executed" denote the action-prediction and execution horizons, respectively.

Task	Demonstrations	RGB views	Predicted	Executed	Eval. trials
Lift Bottle	50	Head	50	50	100
Pull-out Key	50	Head	50	50	100
Lift Can	50	Head + wrist	50	50	100
Put Bottle	50	Head	50	50	100
Insert Hole	50	Head	50	25	100
Insert Tube	50	Head + wrist	50	25	100

Task Selection. The complete UniVTAC benchmark contains eight tasks spanning pose reasoning, shape perception, and contact-rich interaction. Following FTP-1, we omit Grasp Classify and Insert HDMI. Grasp Classify is already saturated by prior tactile baselines, which reach 99–100% success, and therefore provides little resolution for comparing tactile policies. The official Insert HDMI demonstrations are generated by motion planning and complete insertion perfectly, leaving insufficient corrective contact feedback for evaluating whether a policy learns tactile-dependent adjustment. The remaining six tasks cover in-hand and pose-sensitive manipulation (Lift Bottle, Lift Can, and Put Bottle) together with contact-aware insertion and extraction (Pull-out Key, Insert Hole, and Insert Tube) (Yuan et al., 2026a).

Task Definitions and Data. We run the official UniVTAC environments in NVIDIA Isaac Sim 4.5.0 with TacEx 0.1.0, retaining the benchmark's task randomization, demonstrations, and success definitions (Chen et al., 2026a). In Lift Bottle, the policy grasps and raises a bottle while keeping its base within 5 cm of a nearby wall. Pull-out Key randomizes the key's initial rotation and requires the policy to rotate against mechanical resistance before extracting it. Lift Can samples among cylindrical cans of 4, 5, and 6 cm diameter and requires a stable lift without slippage. Put Bottle requires grasping an upright bottle and placing it inside a shelf cavity. Insert Hole presents an inclined hole at either 60° or 120°; the policy must infer its orientation through contact and insert a test tube. Insert Tube requires inserting a 2.0 cm tube into a 2.05 cm opening on an inclined surface. The training set contains 50 automatically collected full trajectories per task, and evaluation uses 100 rollouts with the benchmark's task-specific completion predicates. As in UniVTAC, excessive gel penetration or substantial relative slip violates the physics-based validity conditions rather than constituting successful completion. Figure 8 visualizes the six tasks under this protocol.

C.2 Real-Robot Setup and Task Definitions

The real-robot study covers Play Mahjong, Wipe Board, Pick Up Chips, and Insert USB. These tasks require tactile discrimination, contact-state estimation, force-sensitive grasping, and precise insertion, respectively. Every method uses 50 task-specific demonstrations and is evaluated for 20 rollouts per task. We reset the arm to a recorded task-specific configuration before each rollout and randomize task-relevant object conditions within the demonstrated range.

Hardware and Observations. Our setup comprises a PiPER six-degree-of-freedom arm, a Pika parallel gripper, and two GelSight Mini sensors mounted on opposing fingers (Figure 9). Following the gripper-mounted optical tactile setup used in AnyTouch 2 (Feng et al., 2026), each sensor is attached with a compact two-piece 3D-printed assembly (Figure 10). The policy observes one task-dependent RGB stream, the two synchronized tactile streams, and robot state. Play Mahjong, Pick Up Chips, and Insert USB use a fixed third-person camera, whereas Wipe Board uses a wrist camera to keep the board in view throughout contact. The GelSight cameras stream marker-preserving RGB images at 640×480 and 25 Hz. Before policy inference, RGB observations are resized to 480×270 and tactile observations to 320×240. PiPER provides six arm-joint positions and one



Figure 8: UniVTAC simulation task rollouts. Each row corresponds to one of the six evaluated tasks and shows five key frames illustrating its manipulation and contact progression (Chen et al., 2026a; Yuan et al., 2026a).

gripper coordinate. We map this physical 7-dimensional state to the shared 8-dimensional interface as $[q_1, \ldots, q_6, 0, g]$, where the zero is a dummy seventh arm joint matching the UniVTAC convention. The same dummy dimension is removed from predicted actions before sending commands to PiPER.



Figure 9: Hardware used for real-robot experiments. (a) PiPER six-degree-of-freedom arm. (b) Pika parallel gripper and teleoperation interface. (c) GelSight Mini tactile sensor; two units are mounted on opposing gripper fingers. Product images are from the respective manufacturers.

Demonstrations and Deployment. Demonstrations are collected through the Pika interface at a uniform 20 Hz timeline. Camera, tactile, and joint-state messages are associated by nearest timestamps; samples with excessive inter-stream skew are rejected during dataset preparation. The policy predicts a 50-step chunk in the mixed action representation used by FTP-1 (Yuan et al., 2026a): arm-joint commands are expressed as deltas from the current state, while the gripper command remains absolute. Before execution, the joint deltas are converted back to absolute PiPER joint targets and streamed at 20 Hz. Table 7 summarizes the task-wise execution protocol. Pick Up Chips executes 25 actions before refreshing its observations, whereas the other tasks execute the full 50-step chunk.

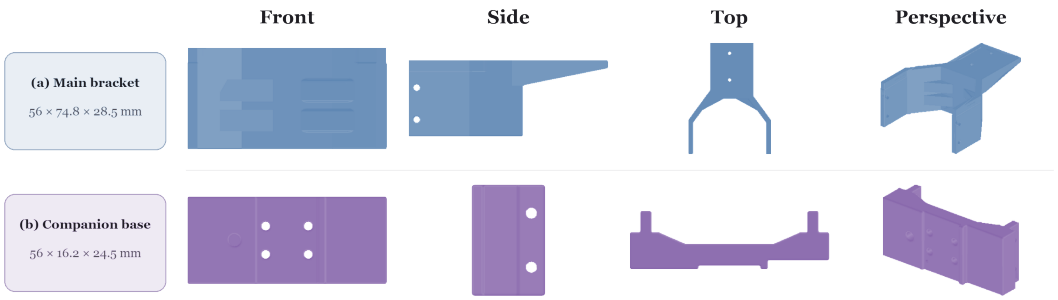

Figure 10: Two-part 3D-printed mount for the GelSight Mini. (a) Main bracket and (b) companion base; the columns show front, side, top, and perspective views. Both parts form one gripper-mounted sensor assembly.

Evaluation rollouts receive no human intervention after execution begins. Emergency stops and infrastructure failures are recorded separately from task failures; the affected rollouts are rerun using the same reset protocol.

Table 7: Real-robot execution protocol for **SimpleTouch**. "Predicted" and "Executed" denote the action-prediction and execution horizons, respectively.

Task	Demonstrations	RGB view	Predicted	Executed	Eval. trials
Play Mahjong	50	Fixed	50	50	20
Wipe Board	50	Wrist	50	50	20
Pick Up Chips	50	Fixed	50	25	20
Insert USB	50	Fixed	50	50	20

Task Protocols. Table 8 summarizes the task-level recipe used for both demonstration collection and evaluation. We vary only conditions that preserve the semantic goal and remain within the demonstrated setup. Success is judged as a binary task outcome at the end of each rollout.

Table 8: Real-robot task protocols. Each policy is trained with 50 demonstrations and evaluated for 20 rollouts per task.

Task	RGB view	Initial-condition variation	Binary success criterion
Play Mahjong	Fixed	The grasped tile is either a bird tile or a one-dot tile; its pose within the gripper is varied. A two-second tactile observation is recorded at the start.	The bird tile is discarded to the table center, or the one-dot tile is placed in front of the robot, followed by a stable release.
Wipe Board	Wrist	Board pose and angle, writing size, and eraser placement in the gripper are varied.	The marked region is wiped clean while the eraser remains securely grasped.
Pick Up Chips	Fixed	The chip and supporting bottle-cap positions are varied.	The chip is grasped and lifted clear without being dropped or visibly broken.
Insert USB	Fixed	USB placement in the gripper is varied while its insertion orientation is kept consistent with the port.	The connector is fully seated in the port and remains inserted after release.

D Baselines and Controlled Variants

D.1 Baseline Methods

We compare with seven policies spanning vision-only imitation learning, pretrained VLA adaptation, visuotactile representation learning, and tactile policy pretraining.

ACT. Action Chunking with Transformers (ACT) (Zhao et al., 2023) is a task-trained visuomotor imitation policy that predicts temporally coherent action chunks from visual observations and proprioceptive state. It receives no tactile input and provides a vision-only reference without policy pretraining.

$\pi_{0.5}$. $\pi_{0.5}$ (Physical Intelligence et al., 2025) is a pretrained VLA policy that couples a PaliGemma visual-language backbone with a flow-matching action expert. In our comparison it is adapted to each task using the available demonstrations but receives no tactile observation, isolating the benefit of adding touch to a strong pretrained VLA.

VITaL. VITaL (George et al., 2025) performs a separate visuotactile pretraining stage before downstream imitation learning. By learning from paired visual and tactile observations, it supplies the policy with an aligned cross-modal representation and represents the explicit-alignment route to tactile policy learning.

UniVTAC-ACT. UniVTAC-ACT (Chen et al., 2026a) is the tactile ACT baseline introduced with the UniVTAC benchmark. It augments the ACT policy with the tactile-centric representation learned by the UniVTAC Encoder, allowing action prediction to condition on both visual and tactile observations.

Tactile-VLA. Tactile-VLA (Huang et al., 2025) connects tactile observations to the physical-interaction knowledge of a pretrained VLA. It combines multimodal fusion with tactile-aware reasoning and hybrid position-force control, providing a task-level route for adapting a pretrained VLA to contact-rich manipulation.

FTP-$\pi_{0.5}$. FTP-$\pi_{0.5}$ (Yuan et al., 2026a) uses the FTP-1 architecture initialized from $\pi_{0.5}$ but omits tactile policy pretraining. Its tactile pathway is learned only from the task demonstrations, making it the closest evaluated baseline for testing whether an existing tactile-policy architecture is sufficient without its large-scale pretraining stage.

FTP-1. FTP-1 (Yuan et al., 2026a) is a generalist tactile policy pretrained on heterogeneous tactile manipulation data spanning multiple sensors and embodiments. It is subsequently adapted to each downstream task and serves as our principal comparison to large-scale tactile policy pretraining.

D.2 Representation and Prediction Ablations

The controlled variants in Figure 4 and Table 3 probe different parts of the recipe. Their definitions are separated in Table 9 so that removing tactile inputs, changing the encoder, and removing prediction supervision are not conflated.

All token-count variants preserve the multi-horizon targets, loss weight, and optimization recipe, so this sweep changes only the spatial granularity of the current tactile context. Adaptive average pooling preserves the global CLS token while reducing the 14×14 patch grid. The vision-only ablation uses the same $\pi_{0.5}$ model and training recipe as the corresponding baseline in Table 1. The only difference is the execution horizon for the insertion tasks: the main-table baseline uses 50 actions, whereas the ablation executes 25 actions before requesting a new observation.

The encoder comparison keeps the expert architecture and predictive objective unchanged. The VAE variant uses the pretrained Wan2.2 VAE checkpoint and its deterministic posterior mean, with the codec kept fixed during policy training. The trainable-T3 variant instead updates only the current-touch encoder path against immutable pretrained-T3 future targets, preventing the auxiliary target itself from drifting. The single- and multi-horizon variants retain frozen T3 features and use the same CLS-plus-6×6 pooled target at each horizon. Single-horizon supervises only $t{+}50$, whereas the full method supervises ten offsets from $t{+}5$ to $t{+}50$, varying temporal supervision while holding per-horizon spatial resolution fixed.

Table 9: Definitions of the reported representation and prediction ablations.

Variant	Controlled change	Exact setting
Vision-only	Remove touch from the policy	Disable the tactile encoder and tactile expert while retaining the task data and $\pi_{0.5}$ action backbone.
Token count	Vary current-touch spatial resolution	Use the T3 CLS token alone (1), or CLS plus adaptively pooled 3×3 (10), 6×6 (37), or 8×8 (65) spatial tokens; 197 retains CLS plus the complete 14×14 grid. Counts are per sensor.
VAE	Replace the tactile-specific representation	Use a frozen Wan2.2 video VAE from Fast-WAM. The two RGB tactile images are concatenated, encoded as one latent frame, and rearranged into 98 tokens of width 192 before projection to the tactile expert. No CLS token or spatial pooling is used.
Trainable T3	Update the tactile encoder	Optimize T3 jointly with the policy from current tactile images; keep future-prediction targets fixed to cached features from the original pretrained T3.
w/o Tac. Pred.	Remove predictive supervision	Set the future-tactile loss weight to zero while retaining the full current-touch pathway and tactile expert.
Single-horizon	Use one future prediction target	Predict CLS plus the same pooled 6×6 spatial grid (37 tokens per sensor) only at $t{+}50$, rather than at ten offsets $t{+}5, \ldots, t{+}50$.
SimpleTouch	Full proposed recipe	Use all 197 current T3 tokens per sensor and multi-horizon future prediction with a 6×6 pooled spatial grid.

D.3 Demonstration-Budget Protocol

Figure 6 compares 10, 20, 50, and 100 task demonstrations for Put Bottle and Insert Hole. For a budget of N, training uses the deterministic prefix of episodes $0, \ldots, N-1$; the subsets are therefore strictly nested. State and action normalization statistics are recomputed from the selected subset, whereas the pretrained $\pi_{0.5}$ and T3 weights and the fixed tactile preprocessing statistics are shared across budgets. The budget counts task demonstrations only and does not include upstream VLA or tactile-representation pretraining data.

We scale optimization duration with the task data: the 10-, 20-, 50-, and 100-demonstration settings train for 4k, 8k, 20k, and 40k updates, respectively, with a global batch size of 64. Warmup remains 5% of training, followed by the same cosine schedule. Model architecture, future-tactile loss, camera protocol, and tail handling remain unchanged. Each point is evaluated with the same task-specific inference protocol as the main results.

D.4 Joint, IDM, and Fast Configurations

The three formulations preserve the same backbone, tactile encoder, tactile expert, multi-horizon targets, loss weights, task data, and repeat-tail supervision. They differ only in whether action generation is conditioned on future tactile variables. Their exact information flow is summarized in Table 10; the measured accuracy and efficiency are reported in Appendix E.3.

For all three modes, current tactile features are prefetched once into a layerwise cache. Future-tactile flow matching uses the same Beta$(1.5, 1)$ time distribution as action flow matching, and both branches use the same multi-horizon T3 targets. This controlled construction attributes the differences in Figure 7 to the prediction formulation rather than to representation capacity or supervision.

Table 10: Training and inference protocols for the three tactile-prediction formulations.

Mode	Training-time action conditioning	Inference
Joint	Action tokens read the K/V cache produced by the same noisy future-tactile branch. Action and tactile flow times are sampled independently, and future-tactile queries never read action tokens.	Future touch and actions are updated together for 10 Euler steps.
IDM	Actions read a separate teacher-forced future-tactile branch. Each sample uses the clean target or, with probability 0.5, an independently noised target; this branch is separate from the auxiliary future-loss branch.	Future touch is denoised for 10 steps, cached at its clean endpoint, and then conditions 10 action-denoising steps.
Fast	The action branch reads only visual-language, state, and current-tactile context. Future tactile prediction is an auxiliary training objective and cannot condition action tokens.	The cached current context directly conditions 10 action-denoising steps; no future tactile latent is generated.

E Additional Analyses and Efficiency

E.1 Additional Future-Tactile Visualizations

Figure 5 compares observed tactile images with PCA visualizations of predicted and ground-truth future tactile latents on Put Bottle and Insert Hole. Figure 11 extends the same analysis to Lift Bottle, Pull-out Key, Lift Can, and Insert Tube, jointly covering contact formation, sustained contact, object motion, and insertion correction. Within each sequence, predicted and ground-truth latents use the same PCA basis fitted to the ground-truth latents, and the displayed color limits are fixed from the ground-truth sequence. Across these task families, the predictions qualitatively follow the principal spatial evolution of contact over multiple horizons rather than an arbitrary auxiliary target.

E.2 Execution Horizon and Feedback Frequency

The policy is trained to predict a 50-action chunk in every task. At deployment, the execution horizon determines how many actions are applied before the policy observes the scene again and replans. Based on task dynamics, we specify H50 for the four non-insertion tasks and H25 for the two insertion tasks before evaluation, without validation- or test-set tuning. Table 11 then evaluates the same task policies under both horizons as a sensitivity analysis; it isolates feedback frequency at inference rather than changing the training target or checkpoint.

Table 11: Effect of execution horizon. Every policy is trained with a 50-action prediction horizon; H25 and H50 execute 25 and 50 actions, respectively, before the next observation. Success rates are reported in percent.

Execution	Lift Bottle	Pull-out Key	Lift Can	Put Bottle	Insert Hole	Insert Tube	Avg.
H50	**100**	**62**	**68**	**50**	58	88	**71.0**
H25	**100**	52	42	39	**85**	**100**	69.7

The preferred execution horizon depends on how quickly useful feedback changes. Lift, extraction, and placement generally benefit from H50: their successful motion is comparatively coherent over the predicted chunk, and executing a longer segment avoids interrupting that motion with unnecessary resampling. Lift Bottle is insensitive in this comparison, whereas the other three non-insertion tasks lose 10–26 points with H25. Insertion reverses the trend. Small changes in contact geometry can alter the required correction direction, so observing touch twice as frequently helps the policy respond before a misalignment becomes a jam, overshoot, or slip. H25 improves Insert Hole and Insert Tube by 27 and 12 points, respectively. These results support the pre-specified task-level rule used in Table 1 without modifying the trained policies.

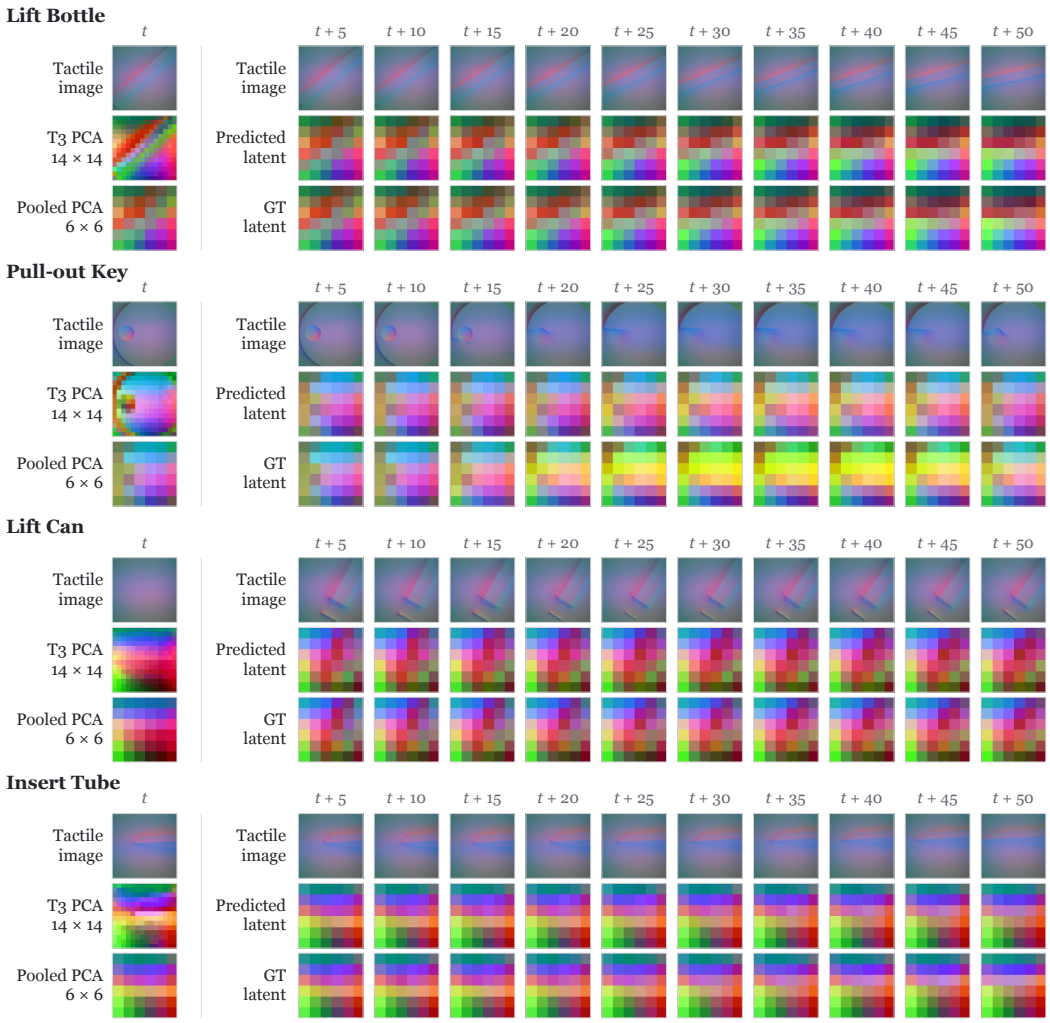

Figure 11: Additional multi-horizon future-tactile predictions for Lift Bottle, Pull-out Key, Lift Can, and Insert Tube. For each task, the current tactile image and its T3 representations appear on the left; subsequent columns compare the observed tactile image, predicted pooled latent, and ground-truth pooled latent at 5-step offsets through $t + 50$.

E.3 Training and Inference Efficiency

Table 12 provides the task-wise measurements underlying Figure 7. Training time is the mean wall-clock time per optimization step with eight H20 GPUs and a global batch size of 64, measured after initialization and data-cache preparation and excluding checkpoint saving. Inference latency is batch-one wall-clock time from synchronized model inputs to one complete action chunk, after warm-up and with GPU synchronization. It includes tactile encoding, context construction, and all model forward passes required by each formulation, and excludes simulator or robot execution and video recording. All methods use ten Euler integration steps, and inference is measured on one A100 GPU.

Fast has the lowest mean training time and inference latency while obtaining the highest mean success. Relative to Joint and IDM, it reduces inference latency by 38.9% and 57.4%, respectively; its mean SR/latency is 1.73 and 2.54 times as high, respectively. The task-level results show that this efficiency does not come from sacrificing contact-sensitive control.

Table 12: Task-wise training and inference efficiency for Joint, IDM, and Fast.

Task	Formulation	Training (s/step)	Latency (s/query)	SR (%)	SR/latency (%/s)
Pull-out Key	Joint	1.891	0.946	56	59.2
	IDM	2.275	1.642	51	31.1
	Fast	1.862	0.614	62	101.0
Put Bottle	Joint	1.890	1.058	46	43.5
	IDM	2.279	1.444	38	26.3
	Fast	1.856	0.632	50	79.1
Insert Hole	Joint	1.898	1.049	80	76.3
	IDM	2.280	1.372	82	59.8
	Fast	1.868	0.603	85	141.0
Insert Tube	Joint	1.897	1.026	99	96.5
	IDM	2.285	1.394	98	70.3
	Fast	1.866	0.643	100	155.5
Mean	Joint	1.894	1.020	70.3	68.9
	IDM	2.280	1.463	67.3	46.9
	Fast	1.863	0.623	74.3	119.1

F Limitations and Future Directions

Our current real-robot evaluation covers parallel-jaw grippers equipped with vision-based tactile sensors; dexterous hands and other tactile sensor types remain unevaluated. Future work should assess whether the approach remains effective across these broader hardware configurations. The present study focuses on tactile understanding, leaving high-frequency tactile feedback and force-based control for future exploration.