TOUCH AS PROGRESS: RESIDUAL POLICY FOR PRECISION ROBOT MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Tactile feedback provides direct access to subtle physical interactions and is an important sensing modality for precision manipulation. Similar tactile signals may occur at different stages of an interaction and call for different actions, leaving the underlying interaction state ambiguous from individual contact observations alone. This work presents **Tactile Progress Residual (TPR)**, which represents the interaction process as progress estimated from recent tactile and robot-state observations and uses it to organize residual correction of base-policy actions. The progress estimator is trained with weak temporal supervision from recorded interactions without manual stage annotations. The estimated progress conditions a lightweight residual policy, provides process-level feedback for learning through progress differences, and determines when residual correction is applied. Conditioned on progress, the residual policy combines tactile feedback with base-policy predictions to generate bounded action corrections. Experiments on ManiFeel, ManiSkill3, and real-world precision manipulation tasks show that TPR consistently outperforms strong vision-only and tactile baselines. Further ablations validate the effectiveness of tactile-based progress estimation and the role of progress in organizing residual correction. Code is released at ananonymous repository.

1 INTRODUCTION

Precision manipulation requires accurate control of physical contact, where small alignment errors can lead to jamming or failure (Fuchioka et al., 2024; Huang et al., 2025). Vision provides useful cues for reaching and positioning but often cannot resolve subtle local interactions. Tactile sensing complements vision with direct contact feedback, making it particularly valuable for precision manipulation (Yuan et al., 2024; Yu et al., 2024; Luu et al., 2025).

However, as illustrated in Fig. 1, similar tactile signals may arise at different stages of an interaction even when the appropriate actions differ (Morihira et al., 2024; Feng et al., 2024). This ambiguity makes it difficult to infer the underlying interaction state from an individual contact observation (Yang et al., 2023; Röstel et al., 2022). As a result, a response suited to one stage may be inappropriate at another, potentially hindering task execution (He et al., 2025; Li et al., 2025).

To incorporate this process context into control, we propose **Tactile Progress Residual (TPR)**, which represents the interaction process as *tactile progress* and uses it to guide a lightweight residual policy over a frozen base agent. Residual control provides a way to refine a pretrained policy's actions through an additional controller (Johannink et al., 2019). In TPR, a progress estimator maps recent tactile and robot-state observations to a continuous interaction-process coordinate, learned through weak temporal supervision from recorded interactions without manual stage annotations (§3.1). The residual policy combines this coordinate with local tactile features, robot state, and base-policy predictions to generate bounded action corrections (§3.2). During residual training, differences in estimated progress supplement the task reward, while the base agent and progress estimator remain fixed (§3.3). At deployment, a threshold on estimated progress determines when the corrections are applied (§3.2), with the base prediction executed directly while the gate is inactive.

This use of tactile progress offers three advantages: **i)** *Process context*: conditioning correction on progress allows the residual policy to interpret local tactile feedback in relation to the ongoing interaction, supporting different corrective responses at different points in the process.

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.

ii) *Process-level feedback*: changes in estimated progress supply an auxiliary learning signal derived from tactile interaction, complementing geometric task rewards with information about how the observed process evolves. **iii)** *Selective intervention*: progress-based gating makes the timing of correction depend on the interaction process, allowing the controller to limit unnecessary interference with the base policy rather than applying corrections throughout execution or whenever contact is detected. Together, these roles make progress a common signal for generating, learning, and scheduling tactile correction, while only a lightweight residual policy requires post-training.

Comprehensive experiments on ManiFeel (Luu et al., 2025) and ManiSkill3 (Tao et al., 2025), together with real-world evaluations on precision contact-rich manipulation tasks, demonstrate the effectiveness of TPR across simulation and physical deployment. TPR consistently improves manipulation success, outperforming the strongest baseline by **11.7** points on ManiFeel, **4.5** points on ManiSkill3, and **13.8** points in real-world experiments. Further ablations and analyses show the benefits of tactile-based progress estimation and residual post-training over a frozen base agent, while highlighting the role of interaction progress in guiding corrective behavior, learning, and intervention during contact-rich manipulation. We release the core code at an anonymous repository.

2 Related Work

Precision and Tactile Manipulation. Precision manipulation requires accurate control of physical interaction, particularly in contact-rich tasks such as insertion, fitting, and assembly, where small errors in alignment or contact can lead to jamming or failure (Fuchioka et al., 2024; Luu et al., 2025; Huang et al., 2025). Tactile sensing complements vision with local interaction cues that are difficult to infer from visual observations alone, and has therefore been increasingly incorporated into manipulation systems. Existing methods directly condition visuomotor policies on measured tactile observations (Yuan et al., 2024; Yu et al., 2024; Luu et al., 2025), use tactile feedback for reactive control or policy refinement (Xue et al., 2025; Huang et al., 2025), or exploit tactile information through predictive or temporally structured models (Zhang et al., 2026b;a; Niu et al., 2026; Wang et al., 2026; Feng et al., 2026; Tian et al., 2026). ContactWorld (Zhang et al., 2026b), for example, incorporates tactile force fields into a vision–tactile world model for contact-dynamics prediction and closed-loop planning, while TacImag (Zhang et al., 2026a) predicts tactile representations from visual and proprioceptive observations to condition the manipulation policy.

TPR incorporates tactile process information through a different modular interface. Without altering the existing base agent, a separate tactile progress estimator summarizes recent tactile-state observations into an interaction-process coordinate. Together with compact local tactile features, the estimated process coordinate conditions a lightweight residual policy and, at deployment, determines when residual correction becomes active. This design preserves the capabilities of the existing base agent while providing tactile guidance when further correction is needed.

Progress Estimation for Robot Manipulation. Progress-aware representations have been increasingly explored in robot manipulation, providing temporal cues beyond instantaneous observations for execution monitoring, subgoal scheduling, action generation, and policy evaluation (Kang & Kuo, 2025; Liu et al., 2026; Zhai et al., 2025; Kim et al., 2026; Dai et al., 2026; Bhardwaj et al., 2026; Yan et al., 2026). TaKSIE (Kang & Kuo, 2025) learns a visual progress representation to adaptively determine when to generate the next visual subgoal. PALM (Liu et al., 2026) models continuous within-subtask progress to maintain temporal consistency and facilitate subtask transitions in long-horizon manipulation. More recently, ProgressVLA (Yan et al., 2026) pretrains an explicit progress estimator on large-scale robot video–text data and leverages it to evaluate predicted future visual states for progress-guided action refinement. These methods show that explicit progress effectively guides robot behavior.

Despite these advances, existing progress-aware methods primarily characterize task evolution from visual, language, or latent representations (Zhai et al., 2025; Kim et al., 2026; Dai et al., 2026; Bhardwaj et al., 2026; Yan et al., 2026). For precision contact-rich manipulation, however, subtle changes in alignment, contact, or constraint can correspond to substantially different physical interaction states while inducing only limited visual variation. Visual progress alone may therefore provide insufficient cues for describing how the interaction evolves after contact. In contrast, TPR estimates an observation-derived interaction-process coordinate from tactile-state observations, capturing how local physical interaction evolves. The estimated coordinate conditions residual cor-

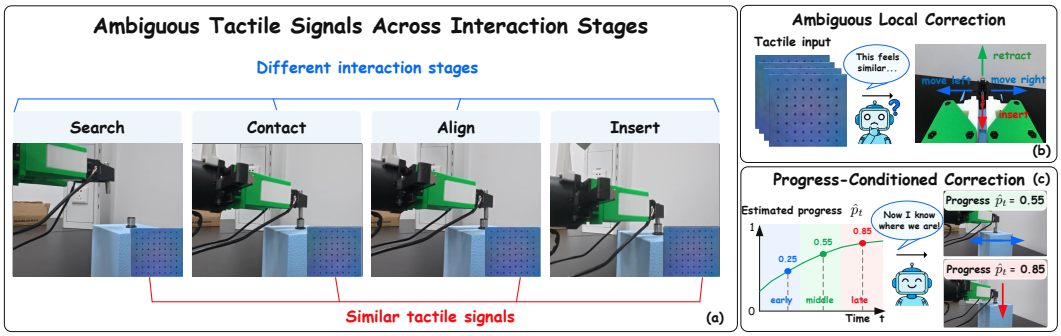


Figure 1: **Motivation.** **(a)** Similar tactile observations can occur at different interaction stages. **(b)** Without process context, these observations may correspond to multiple plausible local corrections. **(c)** Tactile progress disambiguates the interaction process and enables stage-appropriate correction, such as re-centering during alignment and advancing during insertion.

rection, contributes a process-difference signal during residual post-training, and determines when correction is activated at deployment, while keeping the base agent fixed.

3 METHOD

Problem Setup. Let $\mathcal{O}$ denote the base agent's observation space and $\mathcal{A} = [-1, 1]^{d_a}$ its normalized single-action space, where $d_a \in \mathbb{N}_{>0}$ is the action dimension. A pretrained base agent $\pi_0 : \mathcal{O} \to \mathcal{A}^K$ predicts an action chunk of length $K \in \mathbb{N}_{>0}$. We represent the product space $\mathcal{A}^K$ as $K \times d_a$ matrices, with one action per row. At discrete step t, the agent receives $o_t \in \mathcal{O}$ and produces the nominal chunk $\boldsymbol{A}_t^0 = \pi_0(o_t) \in [-1, 1]^{K \times d_a}$, where the superscript 0 denotes the base prediction. The base observation may include vision, proprioception, or touch, depending on the underlying policy. Our goal is to improve contact-rich manipulation through local correction while keeping the pretrained controller fixed throughout residual post-training and deployment.

Overview. As illustrated in Fig. 2, TPR combines a tactile progress estimator with a lightweight residual controller. The estimator summarizes recent tactile-state observations as an interaction-process coordinate $\hat{p}_t \in [0, 1]$ (§3.1). The controller uses this coordinate, local tactile features, robot state, and the base prediction to produce a bounded residual chunk $\boldsymbol{\Delta}_t \in \mathbb{R}^{K \times d_a}$ (§3.2). The resulting action chunk is

$$\boldsymbol{A}_t = \boldsymbol{A}_t^0 + g_t \boldsymbol{\Delta}_t \in [-1, 1]^{K \times d_a}, \tag{1}$$

where $g_t \in \{0, 1\}$ controls residual intervention. Progress thus guides *what* correction to generate, provides an auxiliary reward for learning the correction, and determines *when* to apply it at deployment. Residual post-training and deployment are described in §3.3.

3.1 TACTILE PROGRESS ESTIMATION

Interaction-Process Coordinate. To describe how the observed interaction evolves, we use the causal history $\mathcal{H}_t = [(x_{t-H+1}, \boldsymbol{s}_{t-H+1}), \ldots, (x_t, \boldsymbol{s}_t)]$, where $\mathcal{X}$ is the tactile-observation space, $x_t \in \mathcal{X}$ is the current tactile observation, and $\boldsymbol{s}_t \in \mathbb{R}^{d_s}$ is the observable robot state. Here $d_s \in \mathbb{N}_{>0}$ is the state dimension and $H \in \mathbb{N}_{>0}$ is the history length, so $\mathcal{H}_t \in (\mathcal{X} \times \mathbb{R}^{d_s})^H$.

The progress estimator $f_{\boldsymbol{\theta}}$, parameterized by $\boldsymbol{\theta}$, maps this history to

$$\hat{p}_t = f_{\boldsymbol{\theta}}(\mathcal{H}_t, \kappa) \in [0, 1], \tag{2}$$

where $\kappa \in \{0, \ldots, N_{\text{task}} - 1\}$ identifies the task and $N_{\text{task}} \in \mathbb{N}_{>0}$ is the number of tasks in multi-task estimation. The predicted coordinate describes the relative position of the observation along a recorded interaction process. It is not a success probability, a confidence score, or an independently annotated physical stage of the underlying task execution.

Weak Process Supervision. We train the estimator using relative time within recorded episodes, without manual stage annotations. For an episode of integer length $T > 1$, the progress target is $p_t = t/(T-1) \in [0, 1]$ for $t = 0, \ldots, T-1$, and the endpoint label is $y_t^{\text{end}} = \mathbf{1}[t = T-1] \in \{0, 1\}$,

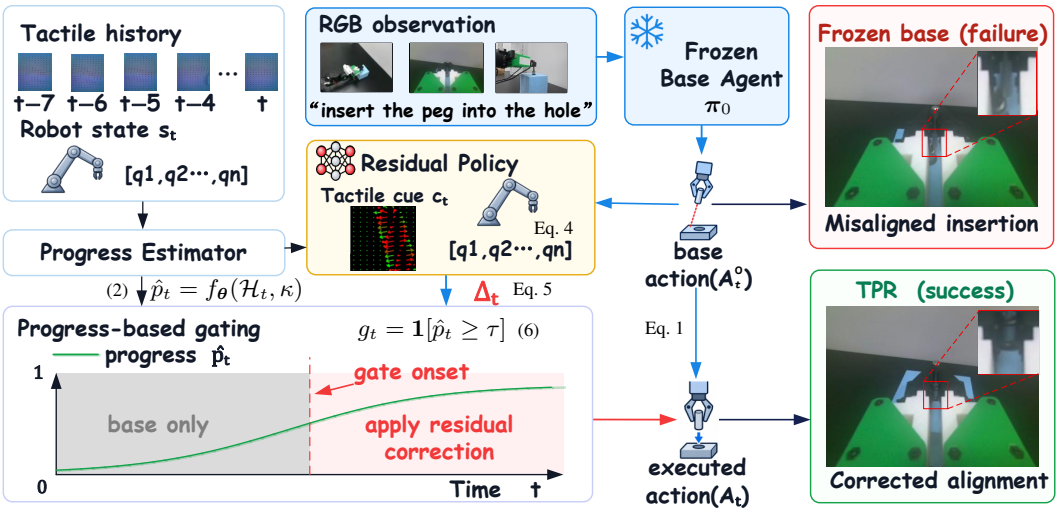


Figure 2: **Overview of Tactile Progress Residual (TPR).** A frozen base agent predicts the nominal action chunk $\boldsymbol{A}_t^0$. The tactile progress estimator summarizes recent tactile and robot-state history into the interaction-process coordinate $\hat{p}_t$. The residual policy combines $\hat{p}_t$, local tactile information, robot state, and the base prediction to generate a bounded correction $\boldsymbol{\Delta}_t$. At deployment, progress-based gating determines whether the residual correction is applied to the nominal action chunk.

where $\mathbf{1}[\cdot]$ is the indicator function. The endpoint marks the end of the recording, including both successful termination and timeout. The final episode length T is used only to construct offline supervision targets and is unavailable during online deployment.

Endpoint prediction and temporal-order classification supplement progress regression. For the latter, each ordered history is paired with its reversed counterpart. Let $\hat{y}_t^{\text{end}}$ denote the endpoint prediction and $\hat{q}_t^{\text{fwd}}, \hat{q}_t^{\text{rev}}$ the forward-order probabilities for the original and reversed histories, respectively, all in $[0, 1]$. The estimator is trained with

$$\begin{aligned} \mathcal{L}_{\text{proc}} &= \mathcal{L}_p + \lambda_{\text{end}}\mathcal{L}_{\text{end}} + \lambda_{\text{ord}}\mathcal{L}_{\text{ord}}, & \mathcal{L}_p &= \text{MSE}\big(\hat{p}_t, p_t\big), \\ \mathcal{L}_{\text{ord}} &= \text{BCE}\big(\hat{q}_t^{\text{fwd}}, 1\big) + \text{BCE}\big(\hat{q}_t^{\text{rev}}, 0\big), & \mathcal{L}_{\text{end}} &= \text{BCE}\big(\hat{y}_t^{\text{end}}, y_t^{\text{end}}\big), \end{aligned} \tag{3}$$

where $\lambda_{\text{end}}, \lambda_{\text{ord}} \geq 0$ weight the auxiliary losses; MSE and BCE denote mean squared error and binary cross-entropy, respectively. Progress regression supervises the full trajectory, while the auxiliary tasks emphasize its endpoint and temporal ordering. Only $\hat{p}_t$ is used for downstream control.

Estimator Architecture. In the ManiFeel implementation, a visual encoder extracts features from each tactile RGB observation. These features are fused with the corresponding robot state and task embedding to form a sequence of tokens. A Transformer aggregates the history, followed by prediction heads for progress, endpoint, and temporal order. The complete network configuration, input dimensions, and task-conditioning details are specified in Appendix B.1.

3.2 Progress-Guided Residual Control

Residual Policy. The residual controller combines local contact information with the estimated interaction progress to refine the nominal action chunk. Its input is

$$\boldsymbol{z}_t = \big[\boldsymbol{s}_{t-1:t}, \text{vec}(\boldsymbol{A}_t^0), \boldsymbol{c}_t, \hat{p}_t\big] \in \mathbb{R}^{d_z}, \tag{4}$$

where brackets denote vector concatenation, $\boldsymbol{s}_{t-1:t} \in \mathbb{R}^{2d_s}$ concatenates two consecutive robot states, and vec flattens the nominal chunk into a vector. The local tactile-contact feature $\boldsymbol{c}_t \in \mathbb{R}^{d_c}$ has dimension $d_c \in \mathbb{N}_{>0}$, giving the input dimension $d_z = 2d_s + Kd_a + d_c + 1$. The contact feature describes contact magnitude, spatial location, bilateral asymmetry, and tactile-image variation without access to target geometry. It provides local contact cues, while $\hat{p}_t$ provides process context.

The actor $\boldsymbol{\mu}_{\boldsymbol{\psi}} : \mathbb{R}^{d_z} \to \mathbb{R}^{d_r}$, parameterized by $\boldsymbol{\psi}$, predicts a residual mean with output dimension $d_r = Kd_a$. During training, it samples a pre-squash residual $\boldsymbol{u}_t \in \mathbb{R}^{d_r}$ from the Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\psi}}(\boldsymbol{z}_t), \text{diag}(\boldsymbol{\sigma}_t^2))$, where $\boldsymbol{\sigma}_t \in \mathbb{R}^{d_r}_{>0}$ is the exploration standard deviation. Squaring

is element-wise, and diag constructs the diagonal covariance matrix. Thus, each selected action component receives a corresponding residual correction.

Bounded Residual Mapping. To keep correction local and the resulting actions valid, we constrain each residual component by both an absolute budget and the available action headroom. Flatten the chunk dimensions and omit the time subscript below. For component $j \in \{1, \dots, d_r\}$, let $a_j^0 \in [-1, 1]$ be the scalar base action, u_j the corresponding component of $\boldsymbol{u}_t$, and $b_j \in [0, 1]$ its residual budget. We compute

$$\begin{aligned} m_j^+ &= \min\big(b_j, 1 - a_j^0\big), & m_j^- &= \min\big(b_j, 1 + a_j^0\big), \\ \bar{\Delta}_j &= \begin{cases} m_j^+ \tanh(u_j), & u_j \geq 0, \\ m_j^- \tanh(u_j), & u_j < 0, \end{cases} & \Delta_j &= \mathrm{clip}\big(\alpha \bar{\Delta}_j, -m_j^-, m_j^+\big), \end{aligned} \tag{5}$$

where $m_j^+, m_j^- \in [0, b_j]$ are the positive and negative headroom and $\alpha \geq 0$ is the correction gain. The operator clip clamps its first argument between the specified lower and upper bounds. Reshaping $[\Delta_1, \dots, \Delta_{d_r}]$ into a $K \times d_a$ matrix gives $\boldsymbol{\Delta}_t$ in Eq. 1. Both $\bar{\Delta}_j$ and Δ_j lie in $[-m_j^-, m_j^+]$, so correction respects the budget even near the action boundaries. Budgets are expressed in normalized action units, rather than physical displacement or rotation.

Progress-Based Intervention. Besides conditioning the residual policy, progress determines whether its correction is applied at deployment:

$$g_t = \mathbf{1}\big[\hat{p}_t \geq \tau\big], \tag{6}$$

where $\tau \in [0, 1]$ is a threshold on the learned process coordinate, not a literal percentage of physical task completion. For $g_t = 0$, Eq. 1 executes the base prediction; for $g_t = 1$, it adds the bounded residual. This gate is used only at inference. During residual post-training, $g_t = 1$ throughout each rollout, while $\hat{p}_t$ continues to condition the residual actor at every replanning step.

Controller Architecture. For ManiFeel, the actor and critic are lightweight MLPs with Tanh activations. The actor output layer is initialized to zero, producing zero deterministic correction at initialization. Network sizes and input–output dimensions are specified in Appendix B.1.

3.3 Residual Post-Training and Deployment

Residual-Only Optimization. We pretrain the progress estimator, then freeze it together with the base agent. Only the residual actor and critic are updated through online PPO. Probability ratios are computed in the pre-squash Gaussian sampling space, and the exploration standard deviation follows a predefined annealing schedule. This decouples progress estimation from residual learning.

Progress-Augmented Reward. The training reward combines the native distance reward, geometric potential shaping, and changes in the frozen progress estimate. Let $d_t \in \mathbb{R}_{\geq 0}$ be the native mean keypoint distance at step t. For a transition from t to $t + 1$, the scalar training reward $r_t \in \mathbb{R}$ is

$$r_t = -d_{t+1} + \beta\big(d_t - \gamma d_{t+1}\big) + \eta\big(\hat{p}_{t+1} - \hat{p}_t\big), \tag{7}$$

where $\beta, \eta \geq 0$ weight the geometric and progress terms, and $\gamma \in [0, 1]$ is the discount factor. The first two terms supply task-directed geometric supervision, while the last term captures temporal changes in the tactile interaction process. The progress difference is an auxiliary reward: unlike the discounted potential form $\gamma\hat{p}_{t+1} - \hat{p}_t$, it carries no assumption of policy invariance. The task geometry used to compute d_t is training-only and excluded from actor and estimator inputs.

Closed-Loop Deployment. At each replanning step, we update the tactile-state history, estimate $\hat{p}_t$, and obtain the nominal chunk $\boldsymbol{A}_t^0$ from the base agent. The residual controller produces $\boldsymbol{\Delta}_t$, which is combined with the base prediction according to Eqs. 1 and 6. We execute the first $h \in \{1, \dots, K\}$ actions before acquiring new observations and replanning. Thus, H controls the estimator's temporal context, K the prediction horizon, and h the execution horizon between replanning steps. Deployment requires neither the final episode length nor task geometry.

For deterministic evaluation, we use a larger correction gain than during PPO post-training, while retaining the same residual budgets and asymmetric headroom clipping in Eq. 5. The gains, budgets, execution horizon, and reward coefficients are specified in Appendix B.1.

Table 1: **Quantitative results on ManiFeel (Luu et al., 2025).** SR across eight precision manipulation tasks, with 250 evaluation episodes per task. Results are reported as mean ± SD. See §4.2 for more details.

Method	Peg ↑	USB ↑	Plug ↑	Gear ↑	Nut ↑	Bulb ↑	Search ↑	Ball ↑	Avg. ↑
Vision Only (Luu et al., 2025)	14	49	51	63	80	76	52	57	55.3
Vision + TacRGB (Luu et al., 2025)	21	49	51	57	78	72	69	72	58.6
Vision + TacFF (Luu et al., 2025)	40	56	58	66	81	81	52	60	61.8
ContactWorld (PC + TacFF)† (Zhang et al., 2026b)	40	50	64	–	25	16	8	18	–
TacImag (Imag. TacFF)† (Zhang et al., 2026a)	31	56	61	59	–	80	–	58	57.5
TPR	**52**±1.5	**62**±1.7	**76**±2.0	**72**±1.5	**86**±1.3	**88**±1.4	**74**±1.8	**78**±1.1	**73.5**±1.1

Table 2: **Quantitative results on ManiSkill3 (Tao et al., 2025).** SR (%) of VLA-based agents on precision manipulation tasks. TPR is evaluated over 250 episodes per task across 10 model seeds. See §4.2 for more details.

Method	Peg ↑	Pick ↑	Plug ↑	StackCube ↑	Avg. ↑
RDT-1B (Liu et al., 2025)	13.2	77.2	1.2	74.0	41.4
OpenVLA (Kim et al., 2024)	0.0	8.0	0.0	8.0	4.0
Octo (Ghosh et al., 2024)	0.0	0.0	0.0	0.0	0.0
Diffusion Policy (RDT) (Chi et al., 2023)	0.0	40.0	0.0	80.0	30.0
TPR	**15.2**±0.8	**84.0**±1.2	**4.4**±0.2	**80.0**±1.2	**45.9**±0.7

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Benchmarks and Metrics. TPR is evaluated on ManiFeel (Luu et al., 2025), ManiSkill3 (Tao et al., 2025), and real-world manipulation to assess progress-aware tactile correction across contact-rich settings. ManiFeel is a visuotactile manipulation benchmark featuring precision tasks where successful execution depends on fine-grained interaction beyond coarse visual alignment. ManiSkill3 (Tao et al., 2025) is further used to evaluate TPR across different types of precision manipulation. Beyond the visuotactile policies considered in ManiFeel, this setting adopts a pretrained VLA model as the base agent, examining whether the proposed progress-aware tactile residual framework remains effective across different policy paradigms. Finally, real-world experiments are conducted on `Peg Insertion`, `USB Insertion`, `Power Plug Insertion`, and `Gear Assembly` with physical tactile sensing. This setting evaluates whether TPR can effectively transfer from simulation to real-world precision manipulation and provide reliable progress-aware tactile correction during physical interaction. Task success rate (SR) is used as the primary metric. For simulation, we report success rate over 250 evaluation episodes per task. The aggregation protocol is benchmark-specific and is detailed in the corresponding experimental and appendix sections.

4.2 SIMULATION BENCHMARK RESULTS

Performance on ManiFeel (Luu et al., 2025). Following the official evaluation protocol (Luu et al., 2025), TPR is evaluated on eight contact-rich precision manipulation tasks. As shown in Table 1, TPR achieves the best performance on all eight tasks, with an average SR of **73.5%**. Compared with the vision-only policy (Luu et al., 2025), TPR improves the average SR by **18.2 points**, demonstrating the benefit of tactile feedback for contact-rich precision manipulation. More importantly, compared with Vision + TacFF (Luu et al., 2025), which incorporates tactile representations as additional policy inputs through feature-level fusion, TPR further improves the average SR by **11.7 points** (**73.5%** *vs.* 61.8%). For methods that exploit tactile information through explicit modeling or prediction of tactile representations, TPR improves the average SR by **42.1 points** over the seven tasks reported by ContactWorld (Zhang et al., 2026b) (**73.7%** *vs.* 31.6%), and by **13.8 points** over the six tasks reported by TacImag (Zhang et al., 2026a) (**71.3%** *vs.* 57.5%). These results show that the advantage of TPR stems not only from introducing tactile feedback, but from using tactile observations to estimate interaction progress and provide progress-aware residual correction, enabling more effective intervention during contact-rich manipulation.

Performance on ManiSkill3 (Tao et al., 2025). Following the ManiSkill evaluation protocol adopted by RDT-1B (Liu et al., 2025), precision manipulation tasks are selected to further evaluate whether TPR remains effective when applied to a pretrained VLA. RDT-1B is used as the frozen base agent. As shown in Table 2, TPR achieves the best overall performance among all compared methods, with an average SR of **45.9%**. Compared with the frozen RDT-1B base agent, TPR improves the average SR by **4.5 points**, showing that tactile residual post-training can effectively

enhance a pretrained VLA without updating its original policy parameters. These results demonstrate that TPR generalizes beyond ManiFeel, with progress-aware tactile correction also improving pretrained VLA agents for precision manipulation across different policy paradigms.

4.3 Real-World Evaluation

Real-world Setup and Protocol. Real-world experiments further evaluate TPR on four precision manipulation tasks: `Peg Insertion`, `USB Insertion`, `Power Plug Insertion`, and `Gear Assembly`. As shown in Fig. 4, experiments are conducted on an ARX X5 robot equipped with a parallel gripper, bilateral Xense G1-WS tactile sensors, a wrist-mounted RealSense D436 camera, and a third-person RealSense D435 camera. During execution, the system receives robot proprioception, wrist-view and third-person visual observations, and bilateral tactile observations. The four tasks span different contact geometries and interaction constraints, all requiring fine-grained local adjustment during contact-rich interaction. The real-world evaluation follows the ManiFeel setting for the base policy, comparison methods, and task-specific success criteria, with the evaluated methods summarized in Table 3. For each task, we collect 100 teleoperated demonstrations to fine-tune the base policy and train the tactile progress estimator. Task success rate (SR) is reported over 100 independent real-world trials for each method and task.

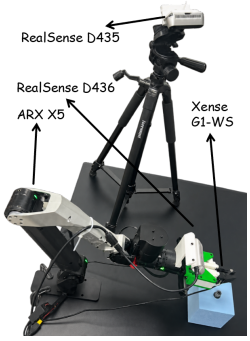


Figure 4: **Real-world setup.**

Quantitative Results. As shown in Table 3, TPR achieves the highest average SR across all tasks, outperforming the strongest baseline (Zhang et al., 2026b) by **13.8 points**. The consistent improvement across insertion and assembly tasks shows that progress-aware tactile residual correction remains effective under diverse physical contact conditions. These results further support tactile progress as process context for local correction in real-world contact-rich manipulation.

Qualitative Visualization. Fig. 3 shows two representative failure-recovery cases on real-world Peg Insertion. In the first pair, both agents contact the hole rim. The non-progress-aware agent slips outward, whereas TPR corrects the lateral misalignment and completes the insertion. The second pair exposes a more challenging ambiguity: pressing against the tabletop produces tactile observations similar to those encountered during insertion. The non-progress-aware agent continues pressing, while progress-conditioned correction enables TPR to retract, re-approach, and succeed on a second attempt. These cases highlight the role of tactile progress: similar contact observations can require different corrections depending on where they occur in the interaction process.

4.4 Diagnostic Experiments

To further examine how interaction progress should be estimated in precision manipulation and whether it is necessary for effective tactile residual correction, we conduct two groups of diagnostic experiments. **First**, we compare visual- and tactile-based progress estimators to determine which modality more reliably reflects the evolution of contact-rich interaction, particularly around the onset and progression of physical contact. **Second**, we study whether explicit progress information is necessary for correction generation, residual learning, and intervention timing by separately removing the progress estimate from the residual-policy input, removing the progress-related term from the training reward, and replacing progress-based gating with contact-triggered activation.

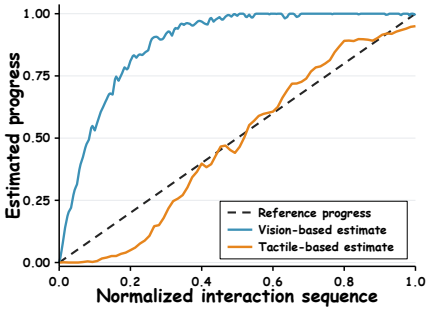


Figure 5: **Progress estimation on a representative unseen ManiFeel rollout.**

Visual vs. Tactile Progress Estimation. Following the ManiFeel setting in §4.2, the proposed tactile progress estimator is compared with a matched visual counterpart. The visual estimator replaces the history of tactile RGB observations with visual RGB observations, while keeping the robot-state history, task conditioning, estimator architecture, and training protocol unchanged. Both estimators are evaluated on held-out ManiFeel episodes unseen during training. Table 4 reports the mean progress-estimation MSE over these episodes, where the tactile-based estimator achieves

Table 3: **Real-world manipulation results.** SR (%) across four precision insertion and assembly tasks, reported over 100 independent trials per method and task. See §4.3 for more details.

Method	Peg ↑	USB ↑	Power Plug ↑	Gear ↑	Avg. ↑
Vision Only	2	28	37	33	25.0
Vision + TacRGB	6	18	31	30	21.3
Vision + TacFF	14	31	42	47	33.5
ContactWorld (PC + TacFF)	12	34	51	39	34.0
TacImag (Imag. TacFF)	7	23	57	41	32.0
TPR	**24**	**41**	**70**	**56**	**47.8**

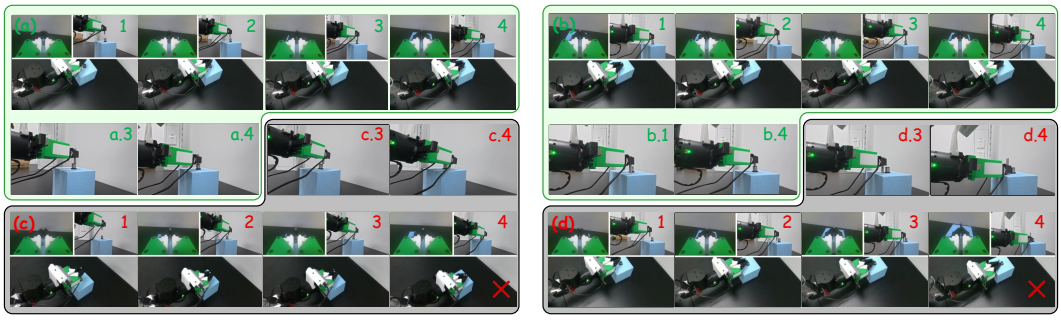


Figure 3: **Qualitative Comparison. (a,c)** Both agents first contact the hole rim; TPR re-centers the peg and completes the insertion, while the non-progress-aware agent slips outward and fails. **(b,d)** Both initially miss the hole and press the peg against the tabletop. Despite similar local tactile observations, TPR uses interaction progress to retract and re-approach, while the non-progress-aware agent continues pressing and fails. Insets highlight the critical interaction stages where the two behaviors begin to diverge.

substantially lower error than its visual counterpart. Fig. 5 further shows a representative rollout. The visual estimator predicts high progress as the robot approaches the precision-manipulation region and saturates well before the interaction is completed. In contrast, the tactile-based estimate evolves more consistently with the reference progress as physical interaction unfolds. These results indicate that visual proximity alone can be ambiguous for estimating interaction progress, whereas tactile feedback provides more reliable cues about the evolution of contact-rich manipulation.

Progress in Policy Input and Reward. Interaction progress enters residual post-training in two ways: as an input to the residual policy and as an auxiliary term in the training reward. To examine these roles separately, we train two ablated variants on ManiFeel. These ablations are evaluated on the four-task subset of Peg, USB, Power Plug, and Gear. *w/o Progress Input* removes $\hat{p}_t$ from the residual-policy input in Eq. 4, while retaining the local tactile-contact feature, robot-state history, and nominal action chunk. *w/o Progress Reward* removes the progress-difference term $\eta(\hat{p}_{t+1} - \hat{p}_t)$ from Eq. 7, while preserving progress conditioning in the residual policy. For both variants, the base agent and progress estimator remain frozen, with the residual architecture, training budget, and evaluation protocol unchanged to ensure a fair and controlled comparison.

As shown in Table 5, removing progress from the residual-policy input reduces the average SR from **66.3%** to **55.5%**. This indicates that explicit progress conditioning provides useful process context for selecting appropriate residual corrections beyond local tactile cues alone. Removing the progress-related reward reduces the average SR to **61.5%**, showing that progress differences also provide a useful learning signal in addition to the geometric task reward. Together, these results show that interaction progress contributes both to correction generation through policy conditioning and to effective residual learning through progress-based reward shaping.

Progress-Based Intervention. To examine the role of progress in intervention timing, we compare the proposed progress-based intervention with contact-triggered activation and always-active correction while keeping the residual policy and evaluation settings unchanged.

Table 6 shows that progress-based intervention achieves an average SR of **73.5%**, outperforming contact-triggered intervention by **8.6** points and always-active correction by **12.7** points. Always-active intervention may introduce unnecessary corrections outside stages where tactile adjustment is needed, while contact-triggered intervention indicates only whether physical contact has occurred, without distinguishing progress within the contact-rich interaction. Progress-based gating instead uses the estimated interaction process to activate residual correction more selectively, limiting un-

Table 4: **Progress-estimation error.** MSE of visual- and tactile-based progress estimation.

Estimator	MSE ↓
Vision-based	0.225
Tactile-based	**0.0052**

Table 5: **Ablation of progress conditioning and reward on ManiFeel (Luu et al., 2025).** SR (%) under different uses of interaction progress during residual post-training. See §4.4 for more details.

Variant	Peg ↑	USB ↑	Power Plug ↑	Gear ↑	Avg. ↑
w/o Progress Input	43	52	67	60	55.5
w/o Progress Reward	48	58	72	68	61.5
TPR	**53**±1.5	**61**±1.7	**77**±2.0	**74**±1.5	**66.3**±1.2

Table 6: **Ablation of intervention on ManiFeel (Luu et al., 2025).** SR under different intervention rules.

Intervention	Peg ↑	USB ↑	Plug ↑	Gear ↑	Nut ↑	Bulb ↑	Search ↑	Ball ↑	Avg. ↑
Always Active	45	50	65	59	74	70	60	63	60.8
Contact Gate	48	53	69	64	77	75	63	70	64.9
Progress Gate	**52**	**62**	**76**	**72**	**86**	**88**	**74**	**78**	**73.5**

necessary interference with the base policy. These results show that contact alone is insufficient for scheduling residual correction; interaction progress provides the process context needed to determine when residual intervention should be activated during contact-rich manipulation.

5 Conclusion

We presented TPR, a progress-aware tactile residual framework for precision contact-rich manipulation. Rather than treating tactile feedback only as an instantaneous observation, TPR estimates a continuous interaction-process coordinate from recent tactile and robot-state history and uses this shared signal to condition residual correction, provide process-level feedback during residual post-training, and determine when correction should be activated at deployment. The resulting design keeps the pretrained base agent fixed while introducing only a lightweight residual controller for local tactile refinement. Across ManiFeel, ManiSkill3, and real-world manipulation, TPR improves average success rate by **11.7**, **4.5**, and **13.8** points, respectively, over the strongest baseline in each setting. Diagnostic experiments further show that tactile observations provide more reliable progress estimates than visual observations in contact-rich interaction, while progress contributes to correction generation, residual learning, and intervention timing. Together, these results support tactile progress as a useful process representation for local correction of frozen robot policies.

AI Use Statement

AI was employed during manuscript preparation for editorial assistance, code troubleshooting, and exploration of figure design. The authors reviewed all AI-assisted material and take full responsibility for the final manuscript.

Reproducibility Statement

The formulation and optimization of TPR, including tactile progress estimation, bounded residual correction, progress-augmented residual learning, and progress-based intervention, are described in § 3. Appendix A provides PyTorch-like pseudocode summarizing the overall training and closed-loop deployment pipeline. Detailed network architectures, input–output dimensions, residual budgets and gains, execution horizons, and reward coefficients are reported in Appendix B. Experimental settings, evaluation budgets, real-world trial counts, and diagnostic protocols are specified in § 4.1–§ 4.4. We release the core implementation and configuration files at an anonymous repository to facilitate reproduction of the reported results and consistent evaluation under the same experimental settings.

REFERENCES

Atiksh Bhardwaj, Edward Weiyi Duan, Prithwish Dan, Wei-Chiu Ma, and Preston Culbertson. Decoding task progress from VLA representations. *arXiv preprint arXiv:2608.13474*, 2026.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin C. M. Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *RSS*, 2023.

Tingjun Dai, Mingfei Han, Tingwen Du, Zhiheng Liu, Zhihui Li, Salman Khan, Jun Yu, and Xiaojun Chang. See, plan, rewind: Progress-aware vision-language-action models for robust robotic manipulation. *arXiv preprint arXiv:2603.09292*, 2026.

Ruoxuan Feng, Di Hu, Wenke Ma, and Xuelong Li. Play to the score: Stage-guided dynamic multisensory fusion for robotic manipulation. In *CoRL*, 2024.

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. In *ICLR*, 2026.

Yuni Fuchioka, Cristian C. Beltran-Hernandez, Hai Nguyen, and Masashi Hamaya. Robotic object insertion with a soft wrist through sim-to-real privileged training. In *IROS*, 2024.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag R. Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *RSS*, 2024.

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.

Binghao Huang, Jie Xu, Iretiayo Akinola, Wei Yang, Balakumar Sundaralingam, Rowland O'Flaherty, Dieter Fox, Xiaolong Wang, Arsalan Mousavian, Yu-Wei Chao, and Yunzhu Li. VT-Refine: Learning bimanual assembly with visuo-tactile feedback via simulation fine-tuning. In *CoRL*, 2025.

Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. In *ICRA*, 2019.

Xuhui Kang and Yen-Ling Kuo. Incorporating task progress knowledge for subgoal generation in robotic manipulation through image edits. In *WACV*, 2025.

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 *CoRL*, 2024.

Seungsu Kim, Jinyoung Choi, Seungmin Baek, and Jean-Michel Renders. ProgVLA: Progress-aware robot manipulation skill learning. *arXiv preprint arXiv:2605.28231*, 2026.

Jinzhou Li, Tianhao Wu, Jiyao Zhang, Zeyuan Chen, Haotian Jin, Mingdong Wu, Yujun Shen, Yaodong Yang, and Hao Dong. Adaptive visuo-tactile fusion with predictive force attention for dexterous manipulation. In *IROS*, 2025.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1B: A diffusion foundation model for bimanual manipulation. In *ICLR*, 2025.

Yuanzhe Liu, Jingyuan Zhu, Yuchen Mo, Gen Li, Xu Cao, Jin Jin, Yifan Shen, Zhengyuan Li, Tianjiao Yu, Wenzhen Yuan, Fangqiang Ding, and Ismini Lourentzou. PALM: Progress-aware policy learning via affordance reasoning for long-horizon robotic manipulation. In *CVPR*, 2026.

Quan Khanh Luu, Pokuang Zhou, Zhengtong Xu, Zhiyuan Zhang, Qiang Qiu, and Yu She. ManiFeel: Benchmarking and understanding visuotactile manipulation policy learning. *arXiv preprint arXiv:2505.18472*, 2025.

Naoki Morihira, Pranav Deo, Manoj Bhadu, Akinobu Hayashi, Tadaaki Hasegawa, Satoshi Otsubo, and Takayuki Osa. Touch-based manipulation with multi-fingered robot using off-policy RL and temporal contrastive learning. In *ICRA*, 2024.

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, Jim Fan, and Trevor Darrell. T-Rex: Tactile-reactive dexterous manipulation. *arXiv preprint arXiv:2606.17055*, 2026.

Lennart Röstel, Leon Sievers, Johannes Pitz, and Berthold Bäuml. Learning a state estimator for tactile in-hand manipulation. In *IROS*, 2022.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-Kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Demonstrating GPU parallelized robot simulation and rendering for generalizable embodied AI with ManiSkill3. In *RSS*, 2025.

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.

Zhenan Wang, Yanzhe Wang, Meixuan Ren, Peng Li, Yang Liu, Yifei Nie, Limin Long, Yun Ye, Xiaofeng Wang, Zhen Zhu, and Huixu Dong. TacMamba: A tactile history compression adapter bridging fast reflexes and slow VLA reasoning. *arXiv preprint arXiv:2603.01700*, 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. In *RSS*, 2025.

Hongyu Yan, Qiwei Li, Jiaolong Yang, and Yadong Mu. ProgressVLA: Progress-guided diffusion policy for vision-language robotic manipulation. *arXiv preprint arXiv:2603.27670*, 2026.

Wenyan Yang, Alexandre Angleraud, Roel S. Pieters, Joni Pajarinen, and Joni-Kristian Kämäräinen. Seq2Seq imitation learning for tactile feedback-based manipulation. In *ICRA*, 2023.

Kelin Yu, Yunhai Han, Qixian Wang, Vaibhav Saxena, Danfei Xu, and Ye Zhao. MimicTouch: Leveraging multi-modal human tactile demonstrations for contact-rich manipulation. In *CoRL*, 2024.

Ying Yuan, Haichuan Che, Yuzhe Qin, Binghao Huang, Zhao-Heng Yin, Kang-Won Lee, Yi Wu, Soo-Chul Lim, and Xiaolong Wang. Robot synesthesia: In-hand manipulation with visuotactile sensing. In *ICRA*, 2024.

Shaopeng Zhai, Qi Zhang, Tianyi Zhang, Fuxian Huang, Haoran Zhang, Ming Zhou, Shengzhe Zhang, Litao Liu, Sixu Lin, and Jiangmiao Pang. A vision-language-action-critic model for robotic real-world reinforcement learning. *arXiv preprint arXiv:2509.15937*, 2025.

Zhiyuan Zhang, Adeesh Desai, Jyun-Chi Hu, Yosuke Saka, Quan Khanh Luu, Jiuzhou Lei, Davood Soleymanzadeh, Bihao Zhang, Minghui Zheng, and Yu She. Imagining the sense of touch: Touch-informed manipulation via imagined tactile representations. *arXiv preprint arXiv:2607.01684*, 2026a.

Zhiyuan Zhang, Pokuang Zhou, Kaidi Zhang, Adeesh Desai, Temitope Amosa, Davood Soleymanzadeh, Jiuzhou Lei, Minghui Zheng, and Yu She. ContactWorld: What matters in vision-tactile world models for contact-rich manipulation. *arXiv preprint arXiv:2606.13877*, 2026b.

SUMMARY OF THE APPENDIX

The appendix provides supplementary algorithmic, implementation, and qualitative details for TPR. It is organized as follows:

- Appendix A presents the PyTorch-like pseudocode for TPR training and closed-loop deployment.
- Appendix B describes the implementation and training configurations of the progress estimator and residual controller.
- Appendix C provides paired success–failure trajectory visualizations and discusses representative failure patterns of TPR.
- Appendix D discusses the limitations of TPR and corresponding future directions.

A PSEUDOCODE

Algorithm 1 summarizes the training and deployment procedure of TPR. The progress estimator is pretrained using weak process supervision and is then frozen together with the base agent. During residual post-training, only the residual actor and critic are optimized. At deployment, the learned progress coordinate determines whether the bounded residual correction is applied.

Algorithm 1 Pseudo-code of TPR in a PyTorch-like style.

```
# pi_0: frozen base agent
# f_theta: frozen progress estimator after pretraining
# mu_psi, V_omega: residual actor and critic
# h: executed actions per chunk; tau: progress threshold

def TRAIN_PROGRESS(D, f_theta):
    #========== Weak process supervision (Eqs. 2--3) ==========#
    for episode in D:
        H, task_id = BUILD_HISTORIES(episode)
        targets = PROCESS_TARGETS(episode)
        pred = f_theta(H, task_id)
        L_proc = PROCESS_OBJECTIVE(pred, targets)
        OPTIMIZER_STEP(f_theta, L_proc)
    return f_theta

def POSTTRAIN_RESIDUAL(pi_0, f_theta, mu_psi, V_omega):
    while TRAINING_CONTINUES():
        o_t, H_t, s_t, x_t, task_id = OBSERVE()

        #========== Progress-conditioned residual (Eqs. 2, 4) ==========#
        p_hat_t = f_theta(H_t, task_id)
        A0_t = pi_0(o_t)
        z_t = RESIDUAL_INPUT(s_t, A0_t, x_t, p_hat_t)

        #========== Bounded correction (Eq. 5) ==========#
        u_t = SAMPLE(mu_psi(z_t))
        Delta_t = BOUNDED_RESIDUAL(A0_t, u_t)
        A_t = A0_t + Delta_t    # g_t = 1 during training
        EXECUTE_PREFIX(A_t, h)

        #========== Residual PPO (Eq. 7) ==========#
        r_t = TRAINING_REWARD()
        STORE_TRANSITION(z_t, u_t, r_t)
        mu_psi, V_omega = PPO_UPDATE(mu_psi, V_omega)

    return mu_psi, V_omega

def TPR(pi_0, f_theta, mu_psi):
    while EPISODE_CONTINUES():
        o_t, H_t, s_t, x_t, task_id = OBSERVE()

        p_hat_t = f_theta(H_t, task_id)
        A0_t = pi_0(o_t)
        z_t = RESIDUAL_INPUT(s_t, A0_t, x_t, p_hat_t)
        Delta_t = BOUNDED_RESIDUAL(A0_t, mu_psi(z_t))

        #========== Progress-gated deployment (Eqs. 6, 1) ==========#
        g_t = INT(p_hat_t >= tau)
        A_t = A0_t + g_t * Delta_t
        EXECUTE_PREFIX(A_t, h)
```

B Implementation Details

This appendix provides benchmark-specific implementation details for TPR. We first describe the configuration used on ManiFeel and then summarize the adaptations used with the frozen RDT-1B base policy on ManiSkill3.

B.1 ManiFeel Configuration

Progress Estimator. Each tactile RGB observation is encoded by an ImageNet-pretrained ResNet-18. The backbone is initially frozen, and only the final residual stage (`layer4`) is fine-tuned. The visual feature is fused with the corresponding robot state and task embedding into a 256-dimensional token. A two-layer Transformer with four attention heads aggregates the history, followed by separate heads for progress, endpoint, and temporal-order prediction. The Transformer feed-forward dimension is 1024 and dropout is 0.1.

The robot-state dimension is $d_s = 7$, the history length is $H = 8$, and the estimator is conditioned on $N_{\text{task}} = 8$ ManiFeel tasks. Histories shorter than H are padded by repeating the earliest available observation. The estimator is trained for 8 epochs with AdamW using a learning rate of 10^{-4}, weight decay of 10^{-4}, and batch size 4.

Residual Controller. For ManiFeel, the normalized action dimension is $d_a = 6$, the base-policy action horizon is $K = 8$, and the local tactile-contact feature dimension is $d_c = 24$. Together with two consecutive robot states and the scalar progress estimate, the residual-policy input dimension is

$$d_z = 2d_s + Kd_a + d_c + 1 = 87,$$

and the residual output dimension is

$$d_r = Kd_a = 48.$$

The actor and critic are two-layer MLPs with 64 hidden units per layer and Tanh activations. The actor output layer is initialized to zero. The residual budgets are 0.25 for translation and 0.10 for rotation in normalized action units.

Post-Training and Deployment. Residual post-training uses PPO with an initial learning rate of 10^{-4}, decayed to 10^{-5}, discount factor $\gamma = 0.99$, and GAE parameter $\lambda = 0.95$. The execution horizon is $h = 4$, so the first four actions of each eight-step chunk are executed before replanning.

The training reward follows Eq. 7,

$$r_t = -d_{t+1} + \beta(d_t - \gamma d_{t+1}) + \eta(\hat{p}_{t+1} - \hat{p}_t),$$

with

$$\beta = 0.5, \qquad \eta = 1.0, \qquad \gamma = 0.99.$$

No explicit success bonus is used.

All reported ManiFeel checkpoints use the hard progress gate in Eq. 6 with $\tau = 0.75$.

B.2 ManiSkill3 Configuration

For ManiSkill3, we use a task-finetuned RDT-1B model as the frozen base policy. RDT-1B predicts a 16-step action chunk, while residual correction is applied only to the first four actions. The residual acts on the 7-dimensional joint-target command, while the gripper action produced by the frozen base policy is left unchanged.

Both progress estimation and tactile-contact processing use a history length of $H = 8$. The residual budget is ± 0.03 rad for each corrected joint target, with correction gain 1. At deployment, ManiSkill3 uses the hard progress gate in Eq. 6 with $\tau = 0.70$.

Residual PPO is performed while keeping the task-finetuned RDT-1B policy frozen. Each reported result is evaluated over 250 episodes, using 10 model seeds evaluated on the same 250 environment seeds. The reported mean and standard deviation are computed across the resulting 10 success rates.

C Additional Qualitative Trajectory Analysis

C.1 Paired Success and Failure Trajectories

We visualize paired successful and failed TPR executions on Gear Assembly, Peg Insertion, Power Plug Insertion, and USB Insertion. Each trajectory is shown at four representative interaction stages: *Approach*, *Align*, *Contact*, and *Outcome*. The large panel at each stage shows the third-person view, accompanied by the first-person observation, tactile force-field representation (TacFF), and tactile RGB observation (TacRGB). These examples provide a qualitative comparison of how successful and failed executions evolve across the four stages.

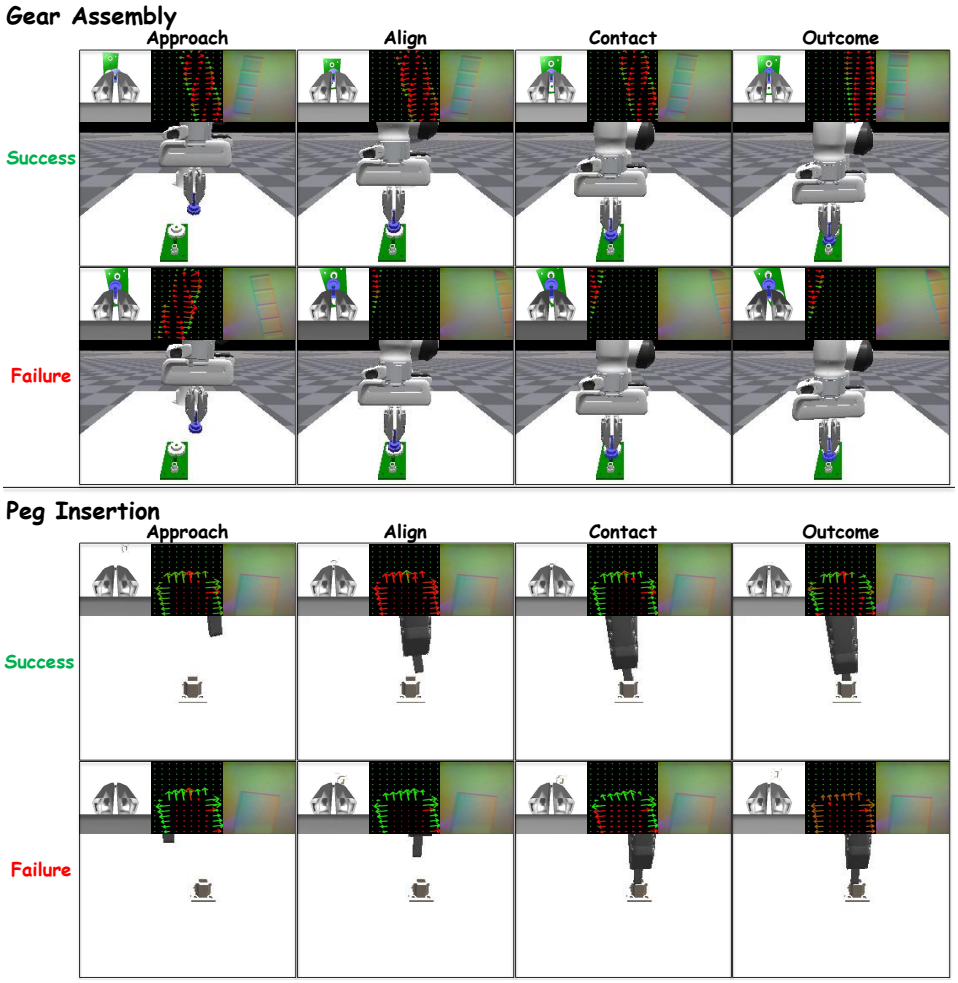


Figure 6: **Paired successful and failed trajectories on Gear Assembly and Peg Insertion.** The successful examples exhibit improved local alignment through the interaction and reach successful engagement or insertion, whereas the failed examples remain misaligned at the final stage.

As shown in Fig. 6, the successful and failed examples can appear similar during the earlier stages and differ more clearly during alignment, contact, and the final outcome. For Gear Assembly, both trajectories reach the task-relevant region, while the successful example achieves engagement and the failed example remains offset. For Peg Insertion, both trajectories approach the receptacle, while the successful example reaches an inserted configuration and the failed example does not.

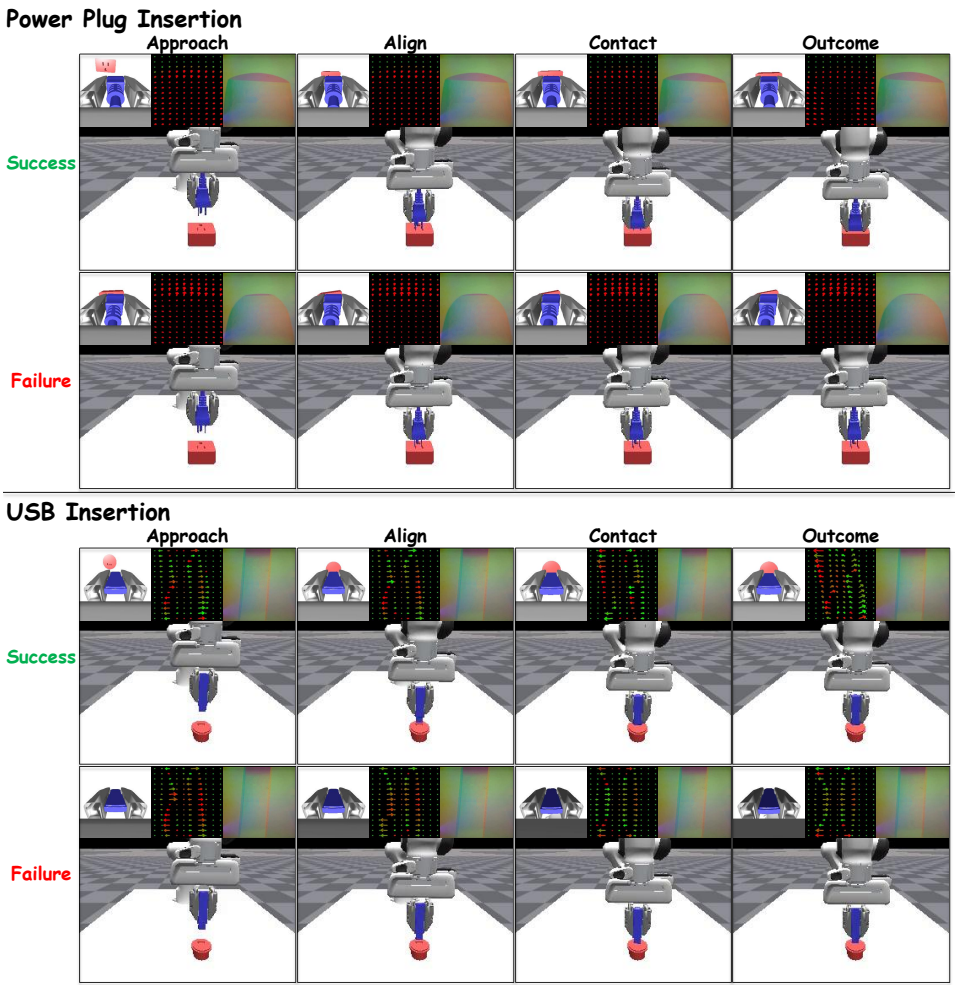


Figure 7: **Paired successful and failed trajectories on Power Plug Insertion and USB Insertion.** The successful examples reach inserted configurations, whereas the failed examples remain locally misaligned or do not complete insertion.

Figure 7 shows similar qualitative differences for Power Plug and USB Insertion. In both tasks, the successful and failed examples approach the target region, while their final configurations differ. The successful examples complete insertion, whereas the failed examples remain outside the completed task configuration. Across the four tasks, the displayed failure examples primarily show two qualitative patterns. First, *persistent local misalignment* is observed when the manipulated object remains offset from the target during or after contact. Second, *incomplete engagement* is observed when the object reaches the target region but does not reach the final inserted or assembled configuration. These examples indicate that reaching the target vicinity alone does not guarantee successful task completion.

D LIMITATIONS AND FUTURE WORK

Limitations. A current limitation of TPR is that the learned progress signal is most informative after meaningful physical interaction begins. Before contact, tactile observations contain limited cues for distinguishing different stages of the approach process, making accurate progress estimation over the entire task considerably more difficult. For this reason, TPR deliberately focuses on the contact-rich phase of precision manipulation: the frozen base policy handles the earlier approach, while progress-based gating activates residual correction only when fine tactile adjustment becomes useful. In addition, the tasks considered in this work start with the target object already grasped by the robot, so the evaluated interaction process does not include object acquisition or grasp formation. While this setting isolates the precision manipulation stage targeted by this work, it also means that TPR has not yet been evaluated as a unified progress model over the full manipulation pipeline, from grasp acquisition to task completion.

Future Work. An important direction is therefore to extend progress estimation from the contact-rich phase to the complete manipulation process. Increasing model capacity and temporal context may improve the representation of longer interaction sequences, while combining tactile feedback with visual and proprioceptive observations could provide informative cues before contact as well as during physical interaction. This could enable future systems to model progress from object acquisition and approach through contact, adjustment, and final task completion. With sufficiently reliable full-trajectory progress estimation, residual correction could be conditioned continuously throughout the task, reducing or ultimately removing the need for an explicit progress-based gate.