Recovery-Aware HIL-SERL for Contact-Rich USB Insertion with Global State Encoding

Anonymous Author(s)
Affiliation
Address
email

Abstract

We study human-in-the-loop real-world reinforcement learning for USB pick-and-insert, a contact-rich manipulation task with tight tolerances and failure modes dominated by contact transitions. Building on HIL-SERL, we deploy an end-to-end on-robot training loop on a UR5e platform and introduce three system-level enhancements that target partial observability and recovery from non-terminal failures: (i) a practical high-rate compliant execution stack for stable in-contact exploration, (ii) global state encoding from an external camera to improve phase awareness under sparse rewards, and (iii) a human-gated **Recovery Buffer** to amplify recovery-critical experience. On real hardware, the global observer accelerates convergence to 100% success from 20k steps (3.5 h) to 12k steps (2 h), and the Recovery Buffer improves recovery success in two representative failure scenarios. We open-source our code and provide a project website with videos and additional details; the link will be released upon acceptance.

1 Introduction

USB connector insertion is a high-frequency and critical operation in electronics manufacturing, automated testing, and equipment maintenance. Improving its automation directly affects throughput, labor cost, and quality consistency. Despite standardized geometry and broad compatibility, USB insertion remains *contact-rich*: it involves tight clearances, strong contact constraints, and sensitivity to interaction forces. Consequently, small pose errors or inappropriate contact forces can cause insertion failure and may damage the connector or the robot end-effector.

Conventional automation pipelines typically rely on precision fixturing, manually tuned impedance/force controllers, or rule-based search strategies. In practice, however, deployment must cope with mounting offsets, compliance of cables and terminals, friction variability, and noisy observations, which makes rule-based solutions brittle and expensive to iterate. Reinforcement learning (RL) offers an attractive alternative by learning control policies through trial-and-error interaction, and has shown strong performance in continuous control and fine manipulation, with representative methods such as PPO and SAC [1, 2]. Yet, for contact-rich tasks, simulation training is highly sensitive to modeling errors in contact dynamics and friction and can suffer from sim-to-real transfer issues. Training directly in the real world removes the transfer gap, but it raises practical challenges: low sample efficiency, high data-collection cost, and non-trivial hardware risks during exploration.

Human-in-the-loop RL (HIL-RL) mitigates these challenges by combining human demonstrations and online intervention with on-robot fine-tuning. Recent progress such as HIL-SERL shows that high-performing real-robot policies can be learned within practical training time [3]. However, when instantiated for USB insertion on a physical UR5e system, we find that two practical bottlenecks remain under-addressed. **First**, the task is *partially observable and strongly temporal*: relying

Submitted to Conference on Deep Learning (CoDL 2025). Do not distribute.

predominantly on local proprioception and end-effector signals often provides insufficient modeling capacity to disambiguate insertion phases and contact transitions, slowing learning and increasing wall-clock cost. **Second**, robust deployment hinges on *learning from mistakes*: exploration yields abundant non-terminal failures (e.g., jamming or misalignment), yet recovery-critical transitions are typically rare and heterogeneous (many-class, few-shot) and are thus easily diluted in generic replay, resulting in weak recovery behaviors and poor robustness to disturbances.

To address these bottlenecks, we study USB2.0 pick-and-insert on a UR5e manipulator and develop an enhanced HIL-RL system that targets *execution safety*, *phase-aware state modeling*, and *recovery-centric learning* while maintaining the HIL-SERL training paradigm. Our contributions are:

- **UR5e deployment with high-rate compliant execution.** We adapt the HIL-SERL baseline to UR5e and implement an end-to-end on-robot training loop with high-frequency command streaming and compliant execution based on PID force/torque regulation, improving stability and feasibility for in-contact exploration.
- **Global stage-aware state encoding for improved task modeling.** We introduce a stage-aware global observer that learns compact task-phase representations from an external camera and fuses them into the policy input, reducing phase ambiguity and accelerating convergence.
- **Human-gated Recovery Buffer for recovery-centric learning.** We add a dedicated Recovery Buffer curated by human gating of recovery-relevant transitions and bias training toward these samples, improving recovery robustness under typical failure states.

The remainder of this paper is organized as follows. Section 2 reviews related work on contact-rich insertion, real-world reinforcement learning, and human-in-the-loop training. Section 3 presents our framework and the three enhancements above. Section 4 evaluates the approach on a UR5e-based USB2.0 pick-and-insert setup, reporting training efficiency, success rate, and robustness under representative failure cases. Finally, Section 5 concludes with discussion and future directions.

2 Related Work

USB insertion falls into *contact-rich* assembly (e.g., peg-in-hole and connector mating), where success depends on tight tolerances, frictional contact transitions, and safe interaction forces. Rather than enumerating methods, we summarize the central trade-offs that motivate our design along three axes consistent with our system (Section 3): (i) compliant execution for safe contact interaction, (ii) observation modeling for phase ambiguity, and (iii) recovery-centric data mechanisms for learning from mistakes.

Compliant assembly and perception-driven insertion. Industrial insertion is often solved with precise fixturing plus impedance/admittance control and handcrafted search heuristics. These approaches can be reliable when uncertainty is bounded, but performance often degrades under mounting offsets, friction variation, and unmodeled compliance. Perception- and servoing-based pipelines improve robustness by grounding insertion in geometric feedback, especially when socket pose changes, but they typically require careful sensing, calibration, and pipeline engineering [4, 5]. Learning-based approaches aim to replace or complement heuristics by extracting contact behaviors from interaction data; prior work shows RL can acquire effective insertion strategies, particularly when paired with compliant/force regulation [6–8]. A recurring lesson is that robust contact-rich performance depends as much on system design—including control interfaces, safety constraints, and reset cost—as on the learning algorithm itself [9]. Our high-rate compliant controller is aligned with this system-first view, providing the execution foundation required for safe real-world exploration.

Real-world RL and human-in-the-loop training. Real-world RL avoids sim-to-real mismatch for contact dynamics, but it makes safe exploration, fast iteration, and robust infrastructure essential. System efforts such as SERL provide an end-to-end software stack for sample-efficient real-robot RL and demonstrate strong performance on contact-rich tasks within short training budgets [10]. HIL-SERL builds on this direction by incorporating human supervision and data collection strategies to further improve efficiency and safety [3]. Complementary work constrains in-contact exploration via shared control and interaction templates, often improving stability but requiring task-specific interface

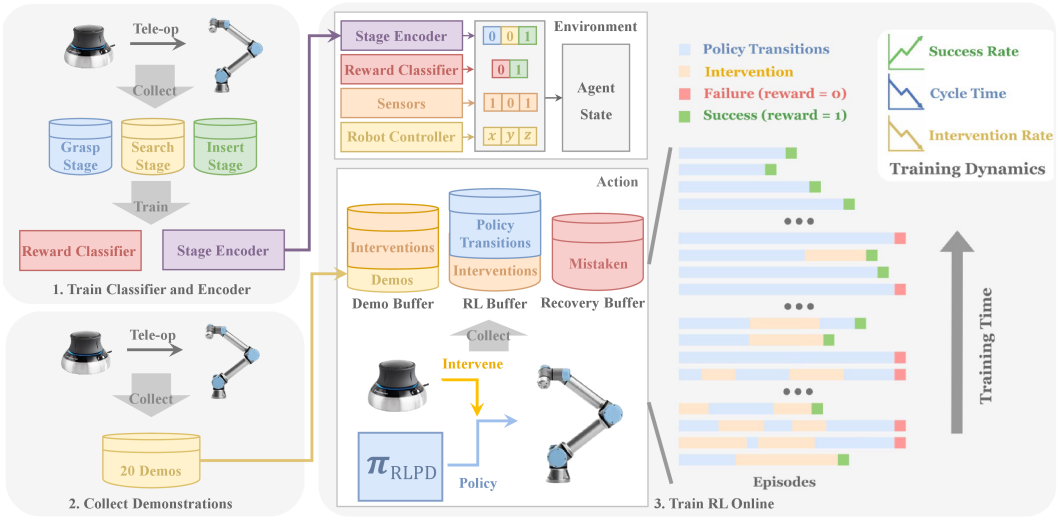


Figure 1: **System overview.** We adopt HIL-SERL as the baseline and augment it with three orthogonal, system-level improvements tailored for real-world USB2.0 pick-and-insert on UR5e. **Execution:** a high-rate compliant controller (RTDE, up to 500 Hz) enables smooth and safe in-contact exploration. **Observation:** a stage-aware global observer encodes external camera images into a compact task-phase representation and fuses it into the policy state. **Data curation:** a human-gated Recovery Buffer explicitly preserves recovery-critical transitions and is sampled together with the Demo Buffer and RL Buffer to strengthen learning from mistakes.

assumptions [11]. In USB insertion, distinct temporal phases (approach, contact, alignment, insertion) amplify partial observability; this motivates augmenting policy inputs with global, phase-aware representations rather than relying solely on local proprioceptive and end-effector signals.

Learning from demonstrations, feedback, and recovery. To reduce exploration burden, learning-from-demonstrations methods integrate expert trajectories into RL through replay and auxiliary objectives [12–14]. Human feedback has also been used to shape behavior online, ranging from evaluative reinforcement (e.g., TAMER) to corrective feedback schemes for deep policies [15, 16]. In contact-rich insertion, however, demonstration-augmented RL can still under-emphasize "near-failure" recovery, and generic replay may not allocate sufficient capacity to sparse recovery transitions. Safety-oriented formulations such as Recovery RL separate task learning from recovery regions/policies to avoid constraint violations [17], but may add algorithmic complexity and require explicit constraint specification. In contrast, our recovery-aware design is lightweight and data-centric: we curate recovery-critical transitions via human gating and bias sampling toward them, strengthening practical robustness without substantially increasing system complexity.

3 Method

Our goal is fast, efficient, and safe real-world RL for contact-rich USB insertion. We start from HIL-SERL [3], which combines sparse reward learning, human interventions, and off-policy training with prior data. We then introduce three targeted enhancements that address the two bottlenecks in Section 2: (i) reliable high-rate compliant execution for safe exploration, (ii) improved task modeling under partial observability via global phase encoding, and (iii) recovery-centric sampling to amplify learning from mistakes.

3.1 HIL-SERL Baseline

We build on Human-in-the-Loop Sample-Efficient Robotic Reinforcement Learning (HIL-SERL) [3], a real-world, vision-based RL system designed to achieve practical training times while maintaining safe on-robot exploration. HIL-SERL uses a sparse reward defined by a learned binary success classifier: the robot is first teleoperated to collect positive and negative examples, which are used to

train a reward detector offline; a small set of human demonstrations is then collected and inserted into an offline *Demo Buffer* to bootstrap policy learning [3]. During online training, the classifier provides the sparse reward signal, while a human operator can intervene via teleoperation (e.g., a SpaceMouse) to correct unsafe or unproductive behaviors; interventions are typically more frequent early in training and are reduced as success rate and cycle time improve [3].

At the system level, HIL-SERL follows an asynchronous actor–learner design. The actor executes the latest policy on the robot, collects transitions, and streams experience to replay, while the learner samples minibatches and periodically pushes updated parameters back to the actor [3]. Experience is stored in two buffers: the Demo Buffer containing offline human trajectories and an *RL Buffer* containing online interaction data. The learner samples *evenly* from these buffers and optimizes the policy with RL with Prior Data (RLPD), instantiated with an off-policy backbone (SAC) [3]. Importantly, intervention segments are treated as privileged corrections: human-controlled transitions are stored in both buffers, while policy transitions around interventions are stored in the RL Buffer, increasing the utility of corrective data for off-policy learning [3]. We adopt this pipeline as our baseline and introduce the following adaptations for USB insertion.

3.2 High-Rate Compliant Controller on UR5e

Contact-rich insertion requires not only sample-efficient learning but also *safe* and *responsive* execution: the control stack must react quickly to contact transitions while remaining compliant to reduce impact forces and hardware risk. To support real-world HIL-RL on UR5e, we implement a high-rate compliant control interface over RTDE (up to 500 Hz). Rather than purely position-based servoing, we employ a PID force/torque regulation layer, which trades some absolute tracking precision for a larger safety margin and stronger disturbance rejection during in-contact exploration. In addition, we drive control via *relative* pose errors to reduce the chance that a policy exploits absolute workspace coordinates.

Concretely, we maintain a global target end-effector pose $(\hat{\mathbf{x}}, \hat{\boldsymbol{\varphi}})$, where $\mathbf{x} \in \mathbb{R}^3$ and $\boldsymbol{\varphi} \in SO(3)$. A lightweight HTTP server asynchronously receives action commands from the agent and updates only this target pose without blocking. A separate high-rate control loop reads the current end-effector pose $(\mathbf{x}, \boldsymbol{\varphi})$ and computes pose errors

$$\delta\mathbf{x} = \hat{\mathbf{x}} - \mathbf{x}, \qquad \delta\boldsymbol{\varphi} = \hat{\boldsymbol{\varphi}} \cdot \boldsymbol{\varphi}^{-1}, \tag{1}$$

where rotational error is computed using $SO(3)$ group operations for geometric consistency. We then generate the desired wrench via a PD law with discrete-time differencing,

$$\mathbf{F} = k_p\delta\mathbf{x} + k_d\Delta(\delta\mathbf{x}), \qquad \mathbf{M} = k_p\delta\boldsymbol{\varphi} + k_d\Delta(\delta\boldsymbol{\varphi}), \tag{2}$$

and apply saturation before sending commands to the robot to enforce safety limits. Since the policy operates at 10 Hz, we interpolate each action into sub-steps and update the target pose at 500 Hz, yielding smoother trajectories and more compliant contact behavior. This controller completes the execution layer needed for stable and safe real-world training.

3.3 Global State Encoding via a Stage-Aware Observer

HIL-SERL learns with a sparse reward defined by a success classifier. While this design is general, it can be inefficient for long-horizon, multi-phase tasks, where informative feedback arrives mainly at task completion and credit assignment across contact transitions becomes difficult. To improve task modeling without changing the sparse-reward paradigm, we introduce a *stage-aware global observer* that encodes external camera images into a compact representation of task phase (Fig. 2).

Stage classifier training. In parallel with training the binary reward classifier, we train an auxiliary stage classifier using global images. For USB insertion, we discretize the task into four stages: *grasp*, *approach*, *hole-search*, and *insertion*. The classifier encodes the global observation with a ResNet-18 backbone [18] and predicts the stage label via an MLP head optimized with cross-entropy. We use the penultimate feature vector as a compact stage representation that summarizes global task context.

Policy integration (stop-gradient). During RL, the stage encoder is frozen and used only for inference: it maps each global image to a 256-dimensional stage representation, which is concatenated to the agent observation. We match this dimension to the baseline visual feature size (256) to keep the representation balanced and avoid over-weighting any single modality.

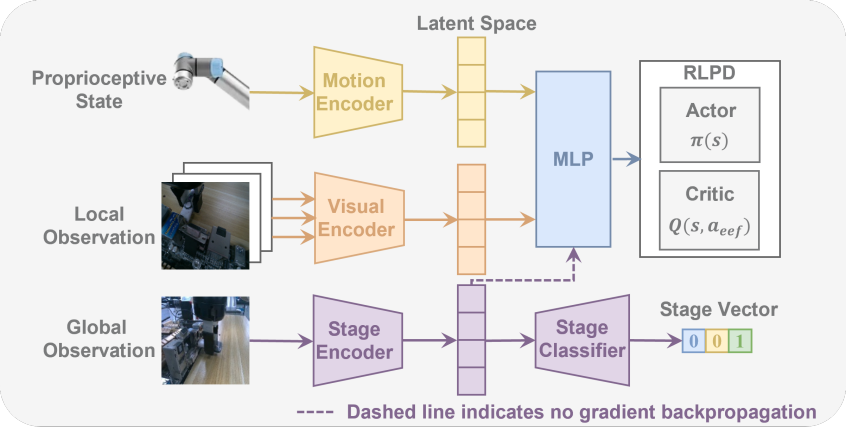


Figure 2: **Global stage-aware state encoding.** We augment the policy observation with a global stage representation extracted from an external camera. A ResNet-18 encoder and an MLP head are trained with cross-entropy to classify four phases (grasp, approach, hole-search, insertion). During RL, we freeze the encoder (stop-gradient) and concatenate its 256-d representation to the policy input to improve phase awareness and temporal task modeling.

Practical impact. Adding the global observer improves phase awareness and temporal modeling. In our UR5e USB insertion setup, this global encoding yields substantially faster convergence (about $1.7\times$ in wall-clock training time) and improves final success (Section 4).

3.4 Recovery Buffer for Learning from Mistakes

Motivated by how humans learn, we explicitly emphasize mistakes: in contact-rich manipulation, recovery from errors often provides more informative supervision than trajectories that simply succeed. HIL-SERL already highlights corrective behavior by prioritizing intervention segments and storing human-takeover transitions in both the Demo Buffer and RL Buffer [3]. However, in USB insertion we find that mistakes are *many-class, few-shot*: there are many distinct error types, but each type appears with limited frequency. Simply mixing these scarce recovery transitions into generic replay therefore dilutes their sampling probability, limiting systematic learning of robust recovery behaviors.

We address this with a dedicated **Recovery Buffer** as an explicit "mistake notebook." At the end of each training episode, a supervisor decides whether the collected transitions contain meaningful recovery behavior (i.e., transitions that correct an error state); if so, the transitions are added to the Recovery Buffer. During learning, we sample from the Demo Buffer, RL Buffer, and Recovery Buffer with a fixed ratio of $4{:}4{:}2$, increasing exposure to recovery-critical experience without changing the underlying RL objective. This lightweight data curation improves recovery behaviors and robustness in our experiments (Section 4).

4 Experiments

We evaluate our system on real-world USB2.0 pick-and-insert with UR5e. The experiments are designed to answer two questions aligned with our method: **(Q1)** does the global stage-aware observer improve sample efficiency and final performance under sparse reward? **(Q2)** does the human-gated Recovery Buffer improve recovery behaviors under typical failure states? Unless otherwise stated, all methods share the same execution stack (Section 3.2) and the HIL-SERL training pipeline (Section 3.1).

4.1 Platform, Task, and Data Collection

Hardware setup. Our platform is a UR5e manipulator equipped with a wrist force/torque sensor and a Robotiq two-finger gripper. We use three Intel RealSense D405 cameras: two wrist-mounted

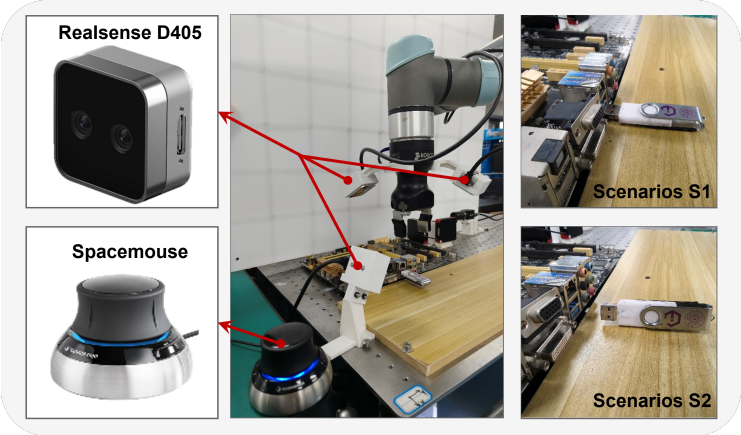


Figure 3: **Platform and failure scenarios. Left:** UR5e USB insertion setup with RealSense D405 cameras and SpaceMouse teleoperation. **Right:** recovery test initializations: **S1** (top) hard-contact under the port; **S2** (bottom) misoriented USB pickup (sideways).

cameras and one side-view camera for global observation. A SpaceMouse is used for teleoperation during data collection and online interventions. The task object is a standard USB2.0 port on a computer motherboard and a white USB flash drive. The policy outputs actions at 10 Hz, while the low-level controller executes at up to 500 Hz via RTDE with compliant force/torque regulation (Section 3.2). Figure 3 (left) provides an overview of the workcell and sensing/teleoperation setup.

Reward classifier and optional stage classifier. Following HIL-SERL [3], we first collect teleoperated rollouts to train a binary success classifier that provides sparse rewards. We collect 200 successful and 600 failed episodes and train a ResNet-10 reward classifier. When enabling global state encoding (Section 3.3), we additionally annotate the collected data with four stages (grasp, approach, hole-search, insertion) and train a stage classifier with a ResNet-18 encoder and an MLP head. The input image resolution is 224×224, and the stage classifier reaches $> 98\%$ validation accuracy within 2–3 minutes.

Demonstrations and online interventions. We collect 20 human demonstrations to initialize the Demo Buffer. During online training, a human operator provides interventions to maintain safe exploration and stabilize early learning. Interventions are scheduled in a curriculum-like manner: initially the operator may complete the full insertion; assistance is then progressively reduced (e.g., only assisting grasp+approach, then only grasp), and finally the policy operates autonomously. We log intervention steps and episode outcomes throughout training.

Evaluation protocol and metrics. We evaluate policies at fixed training checkpoints using 20 independent trials per checkpoint. We report success rate and average insertion time. For learning dynamics, we plot training curves including (i) success rate over training steps and (ii) intervention steps over time. We also report wall-clock time to reach 100% success under our evaluation protocol.

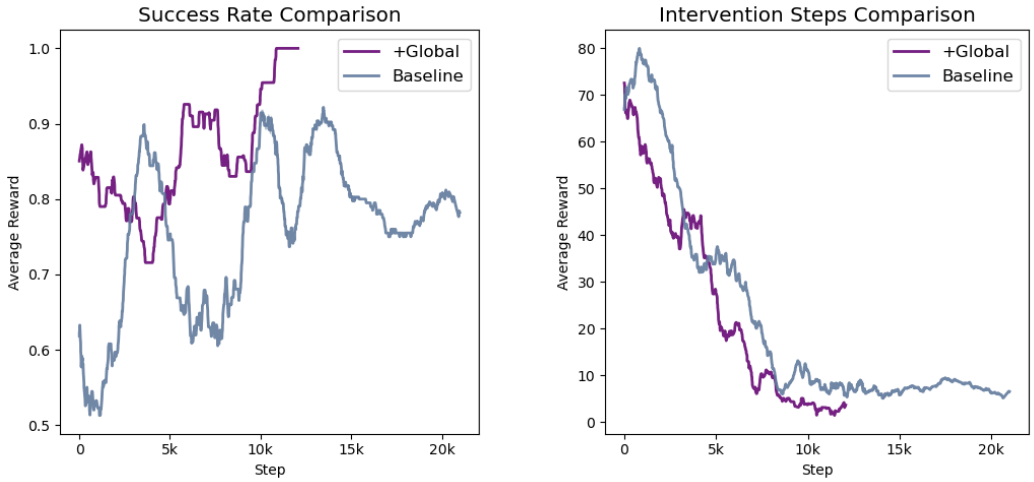
4.2 Global Observer: Convergence and Final Performance

To isolate the effect of the stage-aware global observer, we compare **Baseline** (HIL-SERL with standard observations) vs. **+Global** (Baseline plus global stage encoding; Section 3.3). Both variants use the same reward classifier, demonstrations, intervention schedule, and training setup.

Checkpoint evaluation. We evaluate both methods at 6k, 10k, and 12k training steps. Table 1 reports success rate and average insertion time over 20 trials per checkpoint. Across all checkpoints, **+Global** achieves strictly better performance: it reaches 70% success at 6k steps, while the baseline achieves 0% (no autonomous capability under our protocol). At 10k and 12k steps, **+Global** maintains higher success and consistently shorter insertion time, indicating that phase-aware global encoding not only improves completion but also reduces unproductive contact behaviors.

Table 1: **Global observer evaluation.** Success rate and average insertion time over 20 trials.

Method	6k	10k	12k
Baseline: Success rate (%) ↑	0	75	90
+Global: Success rate (%) ↑	70	95	100
Baseline: Time (s) ↓	30.00	14.60	12.70
+Global: Time (s) ↓	15.15	6.75	5.75



(a) Success rate during training.

(b) Intervention steps during training.

Figure 4: **Learning curves.** Baseline vs. +Global. Global encoding accelerates success and reduces required intervention.

Learning dynamics and wall-clock convergence. Figure 4 provides complementary evidence from the training process. Throughout learning, **+Global** achieves higher success while requiring fewer human intervention steps, indicating improved sample efficiency under the same sparse reward and intervention protocol. Concretely, **+Global** reaches 100% success with near-zero intervention by 12k steps (about 2 h). In contrast, the baseline requires about 20k steps (3.5 h) and still exhibits unstable success and non-trivial intervention demand. Overall, global stage encoding accelerates wall-clock convergence by more than $1.7\times$ in our real-world setting, consistent with the hypothesis that reducing phase ambiguity improves credit assignment and stabilizes contact-rich learning.

4.3 Recovery Buffer: Error Recovery Under Typical Failure States

We evaluate whether the Recovery Buffer improves robustness to error states that arise in contact-rich insertion. We compare **Baseline** vs. **+Recovery** (Baseline plus Recovery Buffer; Section 3.4). During training, **+Recovery** adds human-selected recovery-relevant transitions at episode end and samples Demo/RL/Recovery buffers with a 4:4:2 ratio.

Failure scenarios. We design two representative recovery scenarios, illustrated in Figure 3 (right). **(S1) Hard-contact under the port** (top-right): the USB starts beneath the port in hard contact with the board; successful completion requires retreating to clear contact and re-approaching for insertion. **(S2) Misoriented USB pickup** (bottom-right): the USB is initially placed sideways (tilted on its side) rather than flat on the table, making direct grasp-and-insert infeasible. To succeed, the agent must first recover the object to a graspable pose—by using the gripper to push the USB into the correct orientation—and only then execute the standard pick-and-insert sequence.

Protocol and metrics. For each scenario, we run 20 trials per method starting from the specified failure state. We report recovery success rate and average recovery time.

Table 2: **Recovery evaluation.** Recovery success rate and time (20 trials).

Method	S1: hard-contact		S2: misoriented pickup	
	Success (%) ↑	Time (s) ↓	Success (%) ↑	Time (s) ↓
Baseline	55	10.16	0	30
+Recovery	90	8.63	70	10.05

Results. Table 2 summarizes recovery performance under the two designed failure initializations (Figure 3, right). Adding the **Recovery Buffer** improves both *recovery success* and *efficiency*. In **S1** (hard contact under the port), success rises from 55% to 90% (+35 points) while recovery time drops from 10.16 s to 8.63 s, indicating more reliable retreat–re-align behavior under contact constraints. In **S2** (misoriented USB pickup), the **Baseline** fails to handle the sideways initialization, whereas **+Recovery** learns an explicit error-recovery routine (reorienting the USB by pushing it with the gripper before grasping) and achieves 70% success with a 10.05 s average recovery time. Overall, the gains across both scenarios empirically support our design rationale: prioritizing recovery-relevant transitions in a dedicated buffer strengthens mistake-driven learning without changing the underlying RL objective.

5 Conclusion

This paper presents a recovery-aware human-in-the-loop real-world RL system for contact-rich USB2.0 pick-and-insert on UR5e. Building on HIL-SERL, we introduced three complementary enhancements targeting execution, observation, and data curation: a high-rate compliant control stack for safe in-contact exploration, a stage-aware global observer to reduce phase ambiguity under sparse classifier rewards, and a human-gated Recovery Buffer to amplify learning from mistake-recovery transitions. Experiments on a real UR5e platform show that global stage encoding improves both success rate and insertion time at matched training steps and accelerates end-to-end training to 100% success from 20k steps (3.5 h) to 12k steps (2 h). Moreover, the Recovery Buffer improves recovery robustness in representative failure states, yielding higher recovery success and faster completion time.

Our study highlights that, for contact-rich real-world RL, system design choices—particularly compliant execution, phase-aware state modeling, and recovery-centric data organization—can be as important as the underlying off-policy learner. Limitations include reliance on human supervision for interventions and recovery gating, and evaluation on a single connector family and workcell configuration. Future work will explore reducing human workload via learned gating, extending to broader connector types and tolerances, and integrating richer representations for long-horizon contact sequences. To support reproducibility, we open-source our code and provide a project website with videos and additional details; the link will be released upon acceptance.

References

[1] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL `https://arxiv.org/abs/1707.06347`.

[2] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pages 1861–1870, 2018. URL `https://proceedings.mlr.press/v80/haarnoja18b.html`.

[3] Jianlan Luo et al. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. *Science Robotics*, 10(105):eads5033, 2025. doi: 10.1126/scirobotics.ads5033. URL `https://www.science.org/doi/abs/10.1126/scirobotics.ads5033`.

[4] Rasmus Haugaard, Simon Langaa, Christian Sloth, and Anders Glent Buch. Fast and robust peg-in-hole insertion with continuous visual servoing. In *Proceedings of The 5th Conference on*

Robot Learning (CoRL), volume 164 of *Proceedings of Machine Learning Research*, pages 646–656. PMLR, 2021. URL https://proceedings.mlr.press/v164/haugaard21a.html.

[5] Tao Chen et al. Robust peg-in-hole assembly. In *Robotics: Science and Systems (RSS)*, 2021. doi: 10.15607/RSS.2021.XVII.089.

[6] Takuma Inoue, Giovanni De Magistris, Asim Munawar, and Ryuki Tachibana. Deep reinforcement learning for high precision assembly tasks. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 819–825, 2017. doi: 10.1109/IROS.2017.8202244.

[7] Jianlan Luo, Eugen Solowjow, Chengtao Wen, Juan Aparicio Ojea, and Alice M. Agogino. Deep reinforcement learning for robotic assembly of mixed deformable and rigid objects. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2062–2069, 2018. doi: 10.1109/IROS.2018.8594353.

[8] Cristian Camilo Beltran-Hernandez, Damien Petit, Ixchel G. Ramirez-Alpizar, Takayuki Nishi, Shinichi Kikuchi, Takamitsu Matsubara, and Kensuke Harada. Learning force control for contact-rich manipulation tasks with rigid position-controlled robots. *IEEE Robotics and Automation Letters*, 5(4):5709–5716, 2020. doi: 10.1109/LRA.2020.3010739.

[9] Iñigo Elguea-Aguinaco, Antonio Serrano-Muñoz, Dimitrios Chrysostomou, Ibai Inziarte-Hidalgo, Simon Bøgh, and Nestor Arana-Arexolaleiba. A review on reinforcement learning for contact-rich robotic manipulation tasks. *Robotics and Computer-Integrated Manufacturing*, 81: 102517, 2023. doi: 10.1016/j.rcim.2022.102517.

[10] Jianlan Luo, Zheyuan Hu, Charles Xu, You Liang Tan, Jacob Berg, Archit Sharma, Stefan Schaal, Chelsea Finn, Abhishek Gupta, and Sergey Levine. Serl: A software suite for sample-efficient robotic reinforcement learning. *arXiv preprint arXiv:2401.16013*, 2024.

[11] Abhishek Padalkar, Gabriel Quéré, Antonin Raffin, João Silvério, and Freek Stulp. Shared control templates for in-contact reinforcement learning. *Autonomous Robots*, 48:279–301, 2024. doi: 10.1007/s10514-023-10164-6.

[12] Todd Hester, Mel Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Andrew Sendonaris, Gabriel Dulac-Arnold, Ian Osband, John Agapiou, et al. Deep q-learning from demonstrations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), 2018. URL https://ojs.aaai.org/index.php/AAAI/article/view/11757.

[13] Ashvin Nair, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Overcoming exploration in reinforcement learning with demonstrations. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018. doi: 10.1109/ICRA.2018.8463162. Also available as arXiv:1709.10089.

[14] Bingyi Kang, Zequn Jie, and Jiashi Feng. Policy optimization with demonstrations. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pages 2469–2478. PMLR, 2018. URL https://proceedings.mlr.press/v80/kang18a.html.

[15] W. Bradley Knox and Peter Stone. Interactively shaping agents via human reinforcement: The tamer framework. In *Proceedings of the Fifth International Conference on Knowledge Capture (K-CAP)*, 2009. doi: 10.1145/1597735.1597738.

[16] Rodrigo Pérez-Dattari, Carlos Celemin, Javier Ruiz-del Solar, and Jens Kober. Interactive learning with corrective feedback for policies based on deep neural networks. *arXiv preprint arXiv:1810.00466*, 2018.

[17] Brijen Thananjeyan, Ashwin Balakrishna, Suraj Nair, Michael Luo, Krishnan Srinivasan, Minho Hwang, Joseph E. Gonzalez, Julian Ibarz, Chelsea Finn, and Ken Goldberg. Recovery rl: Safe reinforcement learning with learned recovery zones. *IEEE Robotics and Automation Letters*, 6 (3):4915–4922, 2021. doi: 10.1109/LRA.2021.3070252.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.