2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

SPARR: Simulation-based Policies with Asymmetric Real-world Residuals for Assembly

Yijie Guo[1], Iretiayo Akinola[1], Lars Johannsmeier[1], Hugo Hadfield[1], Abhishek Gupta[2], and Yashraj Narang[1]

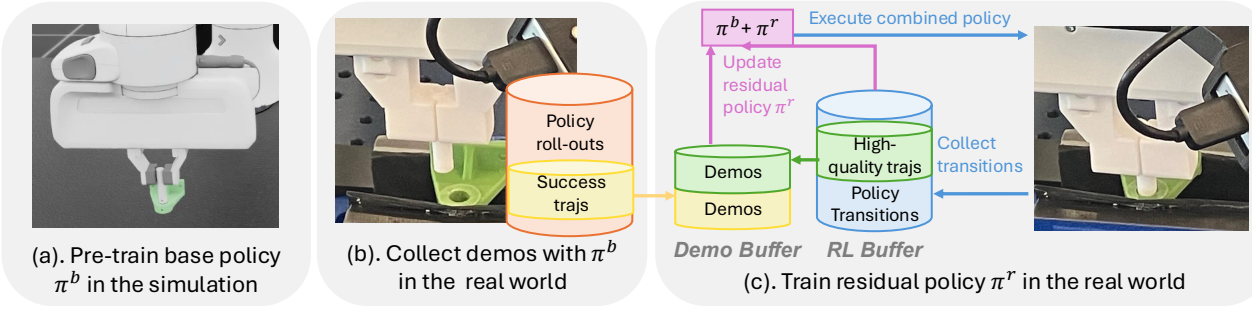

Fig. 1: **Illustration of our approach, SPARR.** (a) A specialist policy is pre-trained in simulation. (b) The simulation policy is deployed zero-shot in the real world, achieving a moderate success rate (e.g., up to 80%). Successful trajectories are collected as demonstrations. (c) A residual policy is trained in the real world on top of the simulation policy, leveraging both the demonstration buffer and the online RL buffer. During training, high-quality trajectories that achieve success quickly are added in demonstrations for further exploitation.

***Abstract*— Robotic assembly presents a long-standing challenge due to its requirement for precise, contact-rich manipulation. While simulation-based learning has enabled the development of robust assembly policies, their performance often degrades when deployed in real-world settings due to the sim-to-real gap. Conversely, real-world reinforcement learning (RL) methods avoid the sim-to-real gap, but rely heavily on human supervision and lack generalization ability to environmental changes. In this work, we propose a hybrid approach that combines a simulation-trained base policy with a real-world residual policy to efficiently adapt to real-world variations. The base policy, trained in simulation using low-level state observations and dense rewards, provides strong priors for initial behavior. The residual policy, learned in the real world using visual observations and sparse rewards, compensates for discrepancies in dynamics and sensor noise. Extensive real-world experiments demonstrate that our method, SPARR, achieves near-perfect success rates across diverse two-part assembly tasks. Compared to the state-of-the-art zero-shot sim-to-real methods, SPARR improves success rates by 38.4% while reducing cycle time by 29.7%. Moreover, SPARR requires no human expertise, in contrast to the state-of-the-art real-world RL approaches that depend heavily on human supervision. Please visit the project webpage at `https://research.nvidia.com/labs/srl/projects/sparr/`**

I. Introduction

Robotic assembly remains a long-standing challenge in robot learning, demanding high-precision, contact-rich manipulation. Simulation and sim-to-real transfer have emerged as powerful strategies for addressing these difficulties. Recent advances in simulation-based learning have led to the development of assembly policies that demonstrate strong performance in both simulated and real world environments [1], [2], [3], [4], [5]. Notably, these methods have achieved success rates of up to 80% on challenging benchmarks of assembly [6], [7]. Despite this progress, current performance is insufficient for industrial deployment, where success rates of 95% or higher are typically required. Furthermore, simulation-trained policies often exhibit brittleness when deployed directly in the real world due to the sim-to-real gap. For state-based policies, zero-shot performance can degrade significantly due to mismatches in physical parameters (e.g., mass, friction), camera calibration errors, state estimation noise, and variations in grasp pose. Vision-based policies, on the other hand, are especially sensitive to visual domain shifts, such as changes in lighting, object appearance, or background, which can severely impair their real-world generalization. Meanwhile, recent progress in real-world RL has demonstrated promising results in contact-rich assembly tasks using raw visual observations [8], [9], [10]. These methods directly optimize policies in real-world environments, enabling them to capture fine-grained physical interactions. However, they typically rely heavily on human expertise for demonstration collection, as well as active supervision and intervention during training to guide learning.

To address these limitations, we propose **SPARR** (**Fig. 1**), a hybrid framework to pre-train a base policy with low-dimensional state observations in simulation and then learn a residual policy with visual observations in the real world. The base policy provides successful demonstrations, a structured prior and safe early exploration, while the residual policy corrects for discrepancies in physical properties, state estimation errors, and visual or environmental differences. This *asymmetric* design enables efficient adaptation to real-world environments without reliance on human supervision.

In the experiments, **SPARR** achieves 95%-100% success rates for two-part assembly tasks (Fig. 2) in the real-world. Compared to the state-of-the-art (SOTA) zero-shot deployment [6], **SPARR** shows a relative improvement of 38.4% in success rate and cost 29.7% less cycle time, averaged over 10 tasks. Unlike the SOTA approach [9], which demands substantial human efforts, **SPARR** requires

[1]Nvidia, [2]University of Washington

no human involvement in real-world learning. In summary, our key contributions are:

- *Asymmetric Residual Learning Framework*: We introduce **SPARR**, a novel framework for sim-to-real transfer, combining a simulation-trained state-based base policy with an asymmetric, vision-conditioned residual policy in the real world. This asymmetric design leverages efficient simulation training while enabling robust adaptation to real-world variations.
- *Autonomous Adaptation Without Human Supervision*: Unlike existing real-world RL methods that require expert demonstrations or frequent human interventions, **SPARR** achieves near-perfect success rates (95–100%) on real-world assembly tasks with zero human supervision, making it practical for scalable deployment.
- *Robustness to State Noise and Physical Variations*: We show that the vision-based residual policy significantly improves robustness to pose estimation errors and socket displacements, outperforming the simulation-trained policy and state-based residual policy.

II. Related Work

a) Robotic Assembly: Robotic assembly is a core challenge in manipulation, requiring accurate perception, precise control under contact, and robustness under uncertainty. Classical approaches based on quasi-static modeling, compliance strategies, and force control have been successful in structured environments [11], [12], but they typically lack adaptability and scalability to new settings. Learning-based methods aim to reduce engineering effort and enable generalization. Recent work uses deep RL, imitation learning, and skill composition to address the contact-rich nature of assembly [5], [4], [6], [7]. Simulation-based learning has made notable progress, but performance often degrades in real-world deployment. Additionally, recent research on real-world RL directly optimizes assembly in the physical world. [13] leverages tactile sensing but requires human expertise to design curriculum specific for insertion tasks. [8] introduces vision-based RL for contact-rich tasks using sparse rewards. [9] and [10] extend this with human-in-the-loop feedback to achieve sub-millimeter accuracy. While these methods push the frontier of real-world robot learning, they often rely on frequent human supervision, intervention, and/or substantial execution time on real-world robots, underscoring the need for more scalable and autonomous solutions. Our work addresses this issue with a hybrid approach that combines the efficiency and structure of simulation policies with residual adaptation in the real world.

b) Sim-to-Real Transfer: Sim-to-real transfer aims to bridge the gap between policies trained in simulation and their deployment in the real world. A common strategy is domain randomization, which exposes the policy to randomized physical and visual parameters during training to improve robustness [14], [15], [16]. Especially for image-based policies, this often requires high-fidelity rendering and large-scale training to identify the effective visual augmentations and randomizations [17]. Yet, such policies can become overly conservative and still fail when the real environment deviates in unmodeled ways. Alternatively, [18], [19], [20] explores domain adaptation, where policies or features are adapted post-training to align with real-world observations. [21] fine-tunes simulation policies for tight-insertion tasks with real-world human demonstrations. Distillation as a widely used adaptation method is time- and compute-intensive [22], [23]. It typically requires significant iterations during real-world development, involving policies rollouts, visual data collection, behavior cloning or DAgger, and fine-tuning.

c) Residual Policy Learning for Assembly Tasks: Residual policy learning has emerged as an effective strategy to enhancce base policies with corrective behaviors, particularly in contact-rich manipulation. Early work [24], [25], [26] augmentes conventional feedback controllers with RL-based residuals to handle unmodeled contact dynamics, demonstrating success in large-part assemblies. [27] learns compliance parameter adjustments via real-time residuals atop a simulation policy. This work supports assembly, pivoting, and screwing tasks though limited to large-part assemblies (with edge lengths exceeding 4 cm). Recently, [28] combines behavior cloning with a residual RL policy, achieving sim-to-real transfer for high-precision assembly, but it requires training both the base and residual policies in simulation as well as collecting human demonstrations. [29] explores learning a vision-based base policy and a force-based residual policy, but primarily evaluates assembly tasks in simulation. [30] proposes compliant residual DAgger with force feedback and compliance control, but relies on human corrections. Across these works, the residual learning paradigm enables compact powerful adaptations but often depend on human expertise for controller design or demonstration collection. In contrast, our approach leverages simulation as a platform to alleviate the need for human demonstrations and is applied to fine-grained assembly tasks (average diameter of plugs $< 1cm$).

III. Method

A. Problem Description

This work studies policy adaptation from simulation to the real world for two-part assembly tasks (Fig. 2). Each environment consists of a Franka robot [31], a *plug* (the part to be inserted), and a *socket* (the corresponding receptacle). The objective is to insert the plug into its designated socket. We formulate the task as a reinforcement learning (RL) problem under the standard Markov decision process (MDP) framework, $M = \{S, A, \rho, P, r, \gamma\}$. At each time step t, $s_t \in S$ is the state observation (e.g., the robot's proprioceptive signals and part poses or visual observations), $a_t \in A$ is the action (e.g., delta end-effector pose), and r_t is the reward. The initial state s_0 is drawn from the distribution $\rho(s_0)$ and transitions follow unknown, possibly stochastic dynamics P. The agent seeks a policy $\pi(a_t|s_t)$ that maximizes the accumulative discounted rewards, and γ is the discount factor.

Our approach is motivated by the practical deployment of robotic assembly policies in industrial and manufacturing settings. A simulation-trained policy typically transfers to the real world with a moderate success rate (e.g., up to 80%

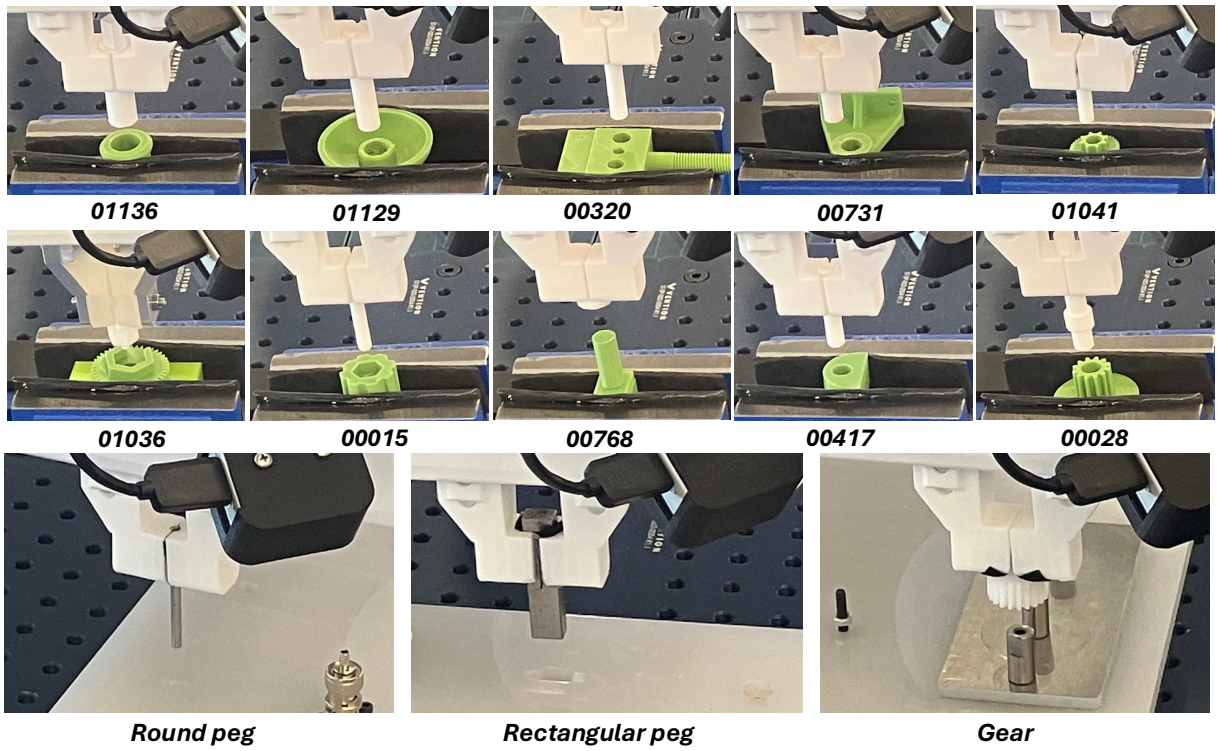

Fig. 2: **Overview of experimental tasks**. (Top) 10 AutoMate [6] assembly tasks. AutoMate provides a dataset of 100 assembly tasks with diverse parts. We choose 10 out of 100 tasks with near-perfect specialist policy pre-trained in simulation. (Bottom) 3 tasks on NIST board. NIST (National Institute of Standards and Technology) provides Assembly Task Boards as performance benchmarks to evaluate robotic assembly technologies. We consider the peg and gear insertion tasks on task board #1.

in [5], [6], [7]). By training a residual policy on a given set of plug-socket pairs in the real world, we aim to boost performance to near-perfect levels. Once trained, the residual policy is combined with the simulation-based base policy and deployed to assemble new plug-socket pairs directly on the assembly line. An overview is shown in Fig. 1. In the following section, we describe base policy pre-training in simulation (Sec. III-B), demonstration collection (Sec. III-D), and residual policy learning in the real world (Sec. III-E).

B. Pre-training in Simulation

As depicted in Fig. 1(a) and Fig. 3, we train a base policy π^b using low-dimensional state-based observations. State-based training in simulation is computationally efficient and robust to variations in object poses and visual patterns when transferred to the real world. In contrast, image-based policies require extensive domain randomization and large-scale training [17] to achieve robustness against changes in dynamics, object poses, and visual domain gaps.

Following AutoMate [6], the leading baseline for zero-shot sim-to-real transfer in assembly, our state observations s_t^b includes robot joint angles, current end-effector pose, goal end-effector pose, and their difference. The action a_t is defined as an incremental pose target, which is tracked by a Cartesian impedance controller. We train the policy π^b with Proximal Policy Optimization (PPO) [32] using imitation rewards derived from disassembly trajectories. π^b outputs Gaussian-distributed actions $a_t \sim N(\mu_t, \sigma_t)$ for continuous control. To enhance robustness, we randomize robot joint configurations, socket poses, and plug poses at the start of each episode. Thanks to reliable state information, dense reward signals, and parallelized simulation environments, the specialist policy can be trained effectively and efficiently to achieve a high success rate in simulation with randomized initial states, even exceeding 99% in some tasks [6].

C. Policy Formulation in the Real World

a) Deployment of Base Policy: When deployed in the real world, π^b takes the end-effector goal pose as part of s_t^b. We obtain the socket pose via a *pose-estimation pipeline* combining Grounding DINO [33], SAM2 [34], and FoundationPose [35]. Then, we set the end-effector goal pose as the estimated socket pose with a z-axis offset, assuming the plug is consistently held in the gripper. For comparison, we also obtain ground-truth goal poses by manually guiding the Franka arm to complete the insertion. On AutoMate assemblies, the difference between our estimated and ground-truth poses is within $\pm 1mm$ along the x and y axes. Based on this observation, we *model state-estimation noise* by uniformly sampling up to $1mm$ in the x and y axes and adding it to the ground-truth goal pose at the start of each episode. We deliberately avoid directly using estimated poses as inputs, which could cause overfitting to the error distribution from our specific pose estimator. For initialization in each episode, the end-effector holding the plug is positioned $2cm$ above the noisy goal pose. In the real world, π^b achieves only a moderate zero-shot success rate due to dynamics mismatch and state-input inaccuracies. We therefore introduce a residual policy π^r to adapt π^b to real-world environments that may differ from pre-training.

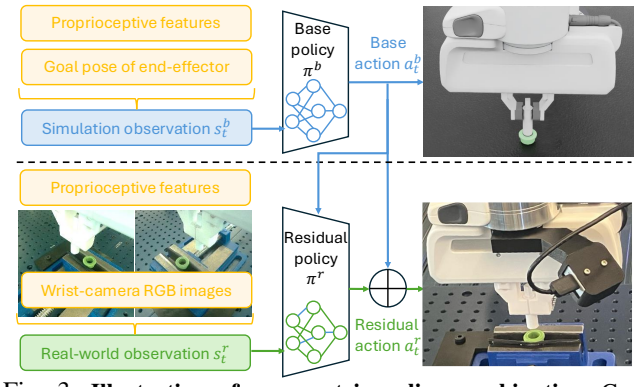

Fig. 3: **Illustration of asymmetric policy combination**. Combining state-based base policies from simulation and image-based residual policies learned in the real-world.

b) Design of Residual Policy: As shown in Fig. 3, π^r takes as input s_t^r, which includes both low-dimensional state (end-effector pose, linear and angular velocities, and force and torques [9]) and visual observations (two RGB images). These inputs *exclude any object pose* and are not affected by state-estimation errors. In addition, visual inputs provide complementary information not captured in the simulated CAD model, such as object asymmetries, textures, manufacturing tolerances and defects, and fine-grained contact features. A CAD-based pose estimator cannot capture geometric constraints such as a USB socket's one-way insertion, whereas visual observations can.

The residual policy π^r outputs actions a_t^r as incremental pose targets, added to a_t^b from π^b. Unlike in simulation, dense reward signals (e.g., disassembly path imitation [6] or signed distance field metrics [5]) are not available in the real world. Also, typical dense-reward terms, such as the negative distance to the goal, can be misleading, since surrounding geometry (e.g., socket walls) may obstruct the plug. We therefore use a sparse success reward: the task succeeds if the current end-effector pose is within $3mm$ in translation and 5° in rotation of the goal, yielding $r_t = 1$.

D. Demonstration Collection in the Real World

Instead of training the residual policy π^r from scratch in the real world, we propose using rollouts of the simulation policy π^b to collect demonstrations for the residual. Since ground-truth residual actions are not available, we generate pseudo-labels by sampling residual actions as Gaussian noise based on the action distribution from π^b. At each timestep t, we gather proprioceptive features with the noisy goal pose as s_t^b and capture camera images for observation s_t^r. The base policy π^b is queried with s_t^b to produce a Gaussian distribution of actions $N(\mu_t, \sigma_t)$. We use the mean μ_t as the deterministic, base action, i.e. $a_t^b = \mu_t$. The residual action a_t^r is then sampled from $N(0, \sigma_t)$. The combined action is defined as $a_t = a_t^b + a_t^r$, which ensures that a_t follows the same distribution $N(\mu_t, \sigma_t)$ as the base policy output.

We record the resulting trajectory as a sequence of transitions $\tau = \{(s_0^r, a_0^b, a_0^r, r_0), \cdots, (s_t^r, a_t^b, a_t^r, r_t), \cdots\}$. As in Fig. 1(b), zero-shot deployment of π^b with injected residual actions yields a subset of successful trajectories, which we store as demonstrations to bootstrap training of π^r.

E. Residual Policy Learning in the Real World

We propose learning a policy π^r to predict residual actions $a_t^r \sim \pi^r(\cdot|s_t^r, a_t^b)$, conditioned on both real-world observations and the base action (Fig. 3). As explained in Sec. III-C, real-world observations s_t^r provide information that is robust to state noise. Conditioning on the base action a_t^b further supplies context for determining the appropriate correction. We empirically evaluate this design choice in Sec. IV-E.

Learning π^r in the real world is challenging due to sparse rewards and limited interaction data. To address these challenges, we adopt the RLPD algorithm [36], which is sample efficient and capable of leveraging prior demonstration data. As shown in Fig. 1(c), at each policy update, transitions are sampled equally from a demonstration buffer and an RL buffer. The demonstration buffer is initialized with the offline success trajectories collected in Sec. III-D. We then continue to update the demonstration buffer with high-quality trajectories encountered during residual policy learning [37]. Specifically, trajectories that achieve task success in fewer time steps than the median of prior demonstrations are added to the buffer, allowing the policy to take advantage of strong experiences gathered during training and decrease cycle time. We further analyze this design choice in Sec. IV-E.

IV. Experiments

In this section, we design experiments to answer the following questions: (1) Can **SPARR** effectively and efficiently adapt simulation policies to real-world assembly tasks with near-perfect success rates, without human demonstrations or interventions? (2) Can **SPARR** achieve robustness to pose variations and pose-estimation errors? (3) Can **SPARR** adapt simulation policies to unseen tasks in the real world?

A. Setup

In our experiments we use the following components:

- A Franka Emika robot [31],
- an NVIDIA Jetson Orin GPU to run robot controllers,
- two Intel RealSense D405 cameras rigidly connected to the flange of the robot,
- a PC with an NVIDIA RTX 4090 GPU to run policies,
- a SpaceMouse input device only used for [9].

B. Effective Real-world Adaptation

We investigate 10 real-world robotic assembly tasks from the AutoMate dataset [6]. We pre-train base policies in Isaac Lab [38] using 128 parallel environments, completing 25 million environment steps within one day of training. We select 10 out of 100 tasks (Fig. 2) that achieve over 99% success in simulation. We choose them for their diverse geometries and strong simulation performance, which are expected to perform well in the real world. We compare following approaches that deploy simulation policies in the real world without human demonstrations:

- **SERL: a real-world RL approach for precise, contact-rich tasks** [8]. Rather than using human demonstrations, we roll out the simulation policy, collect 20 successful trajectories and then run SERL for 0.5 hours to learn real-world assembly policies.

Fig. 4: **Performance on 10 AutoMate tasks**. We evaluate the success rate (↑ higher is better) and cycle time (↓ lower is better) averaged over 20 episodes. SERL, AutoMate, and **SPARR** (Ours) transfer simulation-trained policies to the real world without human effort, where **SPARR** achieves substantially higher success rates and shorter cycle times. HIL-SERL (Oracle) serves as an upper bound, assuming access to near-optimal human demonstrations and continuous human supervision.

- **AutoMate: the SOTA approach of zero-shot sim-to-real transfer for two-part assembly** [6]. We deploy the simulation policy directly in a zero-shot manner.
- **SPARR (Ours)**. We collect 20 successful trajectories with the base policy from simulation but then trains a residual policy on top of the base policy for 0.5 hours.

Additionally, to establish an upper bound on performance, we run **HIL-SERL, the SOTA real-world RL approach** [9] where human experts provide near-optimal demonstrations, frequent interventions, and consistent supervision during training. We manually collect 20 demonstrations with a space mouse and train aseembly policies for 0.5 hours with frequent human interventions to ensure nearly all episodes succeed during real-world training.

We fix the training budget to 0.5 hours across all methods, as real-robot training is expensive and shorter training emphasizes data efficiency. This choice also aligns with recent real-world adaptation and fine-tuning works, which demonstrate effective learning within 30 minutes or less [39], [26]. All actions are executed at 15 Hz. We evaluate policies over 20 episodes with noisy goal poses (Sec. III-C).

In Fig. 4, **SERL** exhibits poor performance due to hard exploration in the sparse-reward setting. With only 20 demonstrations and 0.5 hours of real-world training, it struggles to discover reasonable behaviors and collect positive rewards, as no human intervention is provided during online training. While a few successful trials are observed during real-world learning, SERL cannot efficiently learn to reproduce these successes. **AutoMate** shows a moderate success rate despite strong simulation performance, indicating the sim-to-real gap. In comparison to AutoMate, **SPARR** achieves a relative improvement of 38.4% in success rate and 29.7% in cycle time, highlighting the effectiveness of the residual policy in correcting actions of the simulation policy. Notably, **SPARR** attains a 95–100% success rate, comparable to HIL-SERL, without any human supervision or interventions. **HIL-SERL**, as an oracle approach, shows reduced cycle times compared to other methods that lack human demonstrations or interventions. We attribute this to two main factors: First, sim-to-real transfer approaches use action smoothing and a policy-level action integrator (PLAI) [5] for reliable deployment in the real world, which can negatively affect policy execution speed. Second, the quality of demonstration data differs: faster human demonstrations lead to faster learned policies. Since we constrain real-world training to 0.5 hours per task, RL cannot sufficiently overcome the prior from demonstrations. We leave it to future work to reduce cycle time without increasing training time or human efforts.

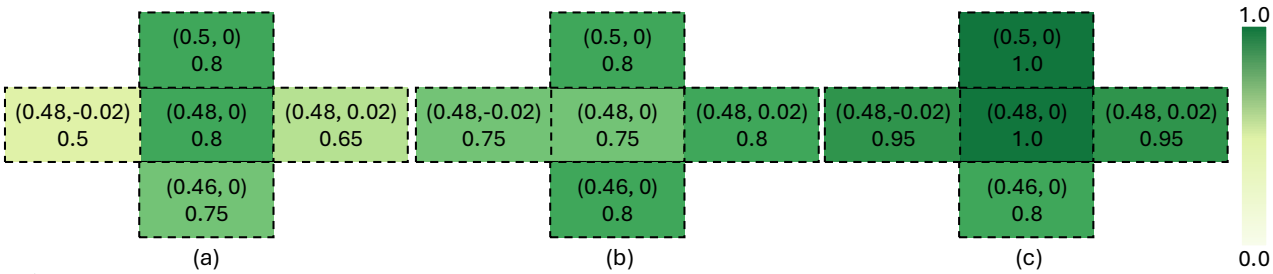

Fig. 5: **Policy deployment on different socket poses for AutoMate task 00731.** Each box indicates the (x, y) coordinates of the socket pose and the corresponding success rate (0–1) during evaluation. The training socket pose is at (0.48, 0), and for evaluation, the socket is displaced by 2*cm*: up (0.50, 0), down (0.46, 0), left (0.48, –0.02), and right (0.48, 0.02). (a) Base policy from simulation. (b) Base policy with a state-based residual policy. (c) **SPARR** (Ours): base policy with image-based residual policy. The color bar represents success rate from 0 (yellow) to 1 (green). **SPARR** achieves higher success rates (darker green) and demonstrates robustness to socket pose variations.

C. Robustness to State Noise

In high-mix, low-volume manufacturing, robustness to part pose variations is a compelling capability. In particular, the socket pose on an assembly line may not exactly match the pose set during residual policy learning. To assess performance under such conditions, we physically displace the socket by 2*cm* from its training position. We also add a uniform noise of 1*mm* to the ground-truth goal pose at each episode, to emulate pose-estimation error, as explained in Sec. III-C. Here two sources of noise are considered: (1) *part pose variation*, due to absence of precise fixtures in flexible automation, and (2) *pose estimation error* since the exact goal pose is unavailable at deployment.

We compare **SPARR** (Ours) with a variant using a state-based residual policy. Our image-based residual policy takes as input two RGB images from wrist cameras, along with the end-effector pose, velocity, force, and torques. The state-based alternative uses the same proprioceptive inputs plus the noisy goal pose instead of images. In Fig. 5, the *state-based residual policy* is sensitive to errors in estimated goal pose and to out-of-distribution socket positions, showing degraded performance. In contrast, our *image-based residual policy* is less affected by socket pose changes, as long as the visual observation remains similar to the training distribution. Its performance only drops slightly under extreme pose shifts, primarily due to the resulting changes in the base action distribution. At novel socket poses, the base policy may produce out-of-distribution base actions that the residual policy cannot fully correct. Overall, **SPARR** with the image-based residual policy achieves a 38.6% improvement over the base policy and outperforms the state-based residual policy by 20.8%, averaged across varying physical socket positions.

D. Generalization to Unseen Tasks

We evaluate the generalizability of **SPARR** on NIST assemblies (Fig. 2) that were not seen during pre-training on AutoMate tasks. We aim to achieve strong real-world performance by adapting the base policy from relevant prior tasks. In Fig. 6, we select the simulation policy based on the object size and the behavior pattern. For 4*mm round peg insertion*, we leverage AutoMate task 01041, which has the smallest plug (6*mm* diameter) among the 10 AutoMate tasks. For 12*mm rectangular peg insertion*, we choose AutoMate task 00731, which has a 10*mm* diameter round peg. For

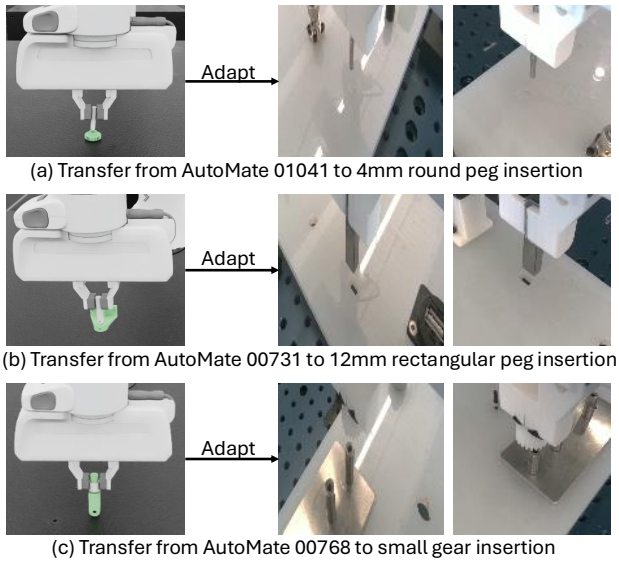

Fig. 6: **Adaptation of simulation policies from AutoMate tasks to NIST tasks.** We show images from wrist-mounted cameras here. Fig. 2 (bottom) shows the NIST tasks from the front camera view.

small gear insertion, we select AutoMate task 00768, whose behavior involves placing a cap onto a cylinder, distinct from other typical peg-in-hole tasks. These choices result in reasonable zero-shot performance on the NIST tasks (success rates of 0.4–0.7). More systematic methods [40] can be applied to reliably identify prior tasks transferring to new tasks according to task similarity and relevance.

When deploying the base policy, differences in dynamics between simulation and reality degrade the performance. However, the policy is largely unaffected by differences in visual observations because it only conditions on low-dimensional state information. As a result, the base policy still generates some successful demonstrations and serves as a functional prior for real-world training. We then train a residual policy using **SPARR** to improve real-world success.

Fig. 7 shows the generalizability of **SPARR** under unseen dynamics and visual observations. For *round peg insertion*, **SPARR** achieves only an 80% success rate because the tiny hole on the reflective white board is extremely difficult to detect (Fig. 2&6). Although the goal is unclear in the image observation, the residual policy still helps to improve performance using input from the proprioceptive state. For *rectangular peg insertion*, the base policy was trained to

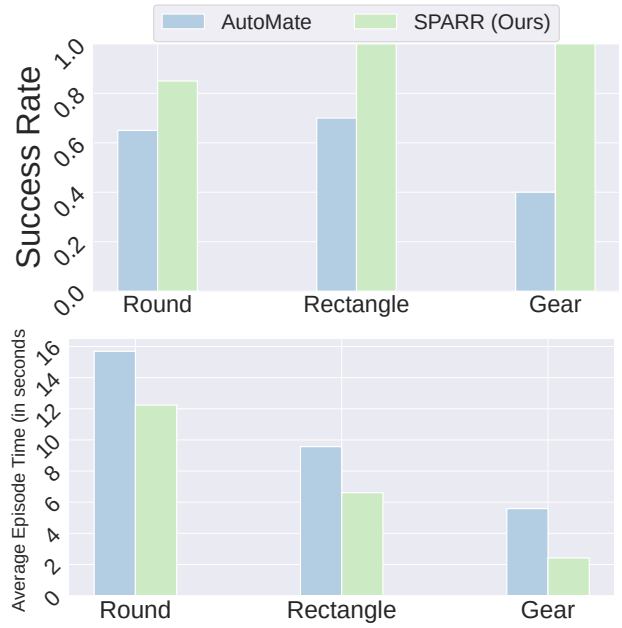

Fig. 7: **Performance on NIST assembly tasks. SPARR** outperforms the baseline in success rate (↑ higher is better) and cycle time (↓ lower is better).

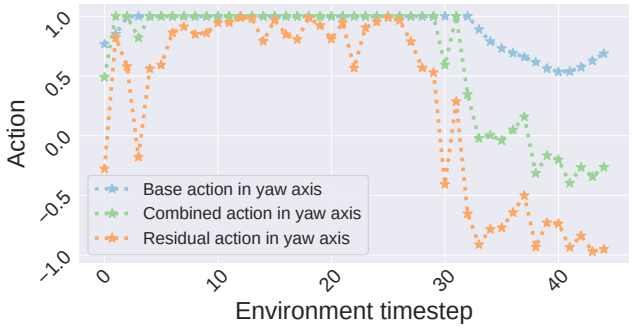

Fig. 8: **Visualization of base, residual and combined actions.** On a trajectory completing rectangular peg insertion task, the residual action in yaw axis disagrees with the base action and changes the plug rotation when the combined action is executed in the environment. All these actions are clamped to the range [-1,1] as incremental target pose.

insert a round peg and cannot handle the slight rotational mismatch with the rectangular hole. To emphasize this challenge, we add a uniform yaw noise of up to 3 degrees to the end-effector pose at the start of each episode. Our residual policy learns the necessary rotation to align the peg with the target hole (Fig. 8), achieving a perfect success in all 20 evaluation episodes. For *small gear insertion*, the gear clearance is much tighter ($0.005 - 0.014mm$) than the plug in AutoMate task 00768 ($0.5 - 1mm$). The residual policy learns to correct both the insertion angle and the applied actions, resulting in successful insertions. Overall, **SPARR** achieves a relative improvement of 74.5% in success rate and 36.5% in cycle time across these NIST tasks.

E. Ablation Study

a) Effect of demo buffer update: As described in Sec. III-E, we update the demonstration buffer during RLPD policy learning. Specifically, we add high-quality trajectories that reach success faster than at least half of the offline demonstrations. This allows the policy to retain and exploit valuable experiences encountered during training, enabling it to complete the task faster. Table I compares methods with and without this demo buffer update, showing that **SPARR** outperforms the variant that does not update the buffer.

TABLE I: **Comparison of method variants. SPARR** shows higher success rate than the variant without demo buffer update.

	Success rate (↑)		Cycle time (s) (↓)	
Task	01041	01036	01041	01036
AutoMate	0.8	0.8	10.76	4.82
SPARR w/o demo update	0.95	0.8	8.1	**4.31**
SPARR (Ours)	**1.0**	**1.0**	**6.81**	5.45

b) Effect of base action as input to residual policy: Including the base action as an input provides important context for the residual correction. In Table II, without the base action as input, the residual policy still slightly improves zero-shot deployment performance of AutoMate. compared to this variant without base action as input, **SPARR** further enhances both the success rate and the cycle time.

TABLE II: **Comparison of method variants. SPARR** outperforms the variant without base action as input to residual policy.

	Success rate (↑)		Cycle time (s) (↓)	
Task	00015	00768	00015	00768
AutoMate	0.75	0.65	7.66	10.19
SPARR w/o base action input	0.8	0.8	6.12	8.76
SPARR (Ours)	**1.0**	**0.95**	**3.51**	**6.11**

V. Conclusion and Future Work

In this work, we propose a residual policy learning approach **SPARR** that leverages a simulation-trained state-based policy as a base and augments it with an asymmetric, vision-conditioned residual in the real world. The base policy provides structured priors and guides exploration, while the residual corrects for real-world discrepancies. On real-world two-part assembly tasks, **SPARR** achieves near-perfect success without requiring human demonstrations or interventions, while also demonstrating robustness to state noise and generalizability to unseen tasks.

While **SPARR** is highly effective, it relies on several assumptions. First, it requires the plug to be rigidly pre-grasped in the gripper. Investigating the full grasp-to-insertion pipeline and improving robustness to grasp perturbations remain topics for future work. Second, **SPARR** depends on reasonably good zero-shot sim-to-real transfer of the base policy. If the base policy achieves near-zero success in the real world, the residual component alone is insufficient to recover performance. Finally, **SPARR** assumes access to automated reward or success detection during real-world deployment. Relaxing these assumptions is an important direction for future research. It would be valuable to explore more expressive residual policies and more reliable success classifiers, such as multimodal models that integrate visual, force, and audio signals. Extending the framework to more diverse and challenging tasks beyond insertion-style assemblies is another promising direction.

REFERENCES

[1] G. Thomas, M. Chien, A. Tamar, J. A. Ojea, and P. Abbeel, "Learning robotic assembly from cad," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3524–3531.

[2] Y. Fan, J. Luo, and M. Tomizuka, "A learning framework for high precision industrial assembly," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 811–817.

[3] Y. Narang, K. Storey, I. Akinola, M. Macklin, P. Reist, L. Wawrzyniak, Y. Guo, A. Moravanszky, G. State, M. Lu, *et al.*, "Factory: Fast contact for robotic assembly," *arXiv preprint arXiv:2205.03532*, 2022.

[4] Y. Tian, K. D. Willis, B. Al Omari, J. Luo, P. Ma, Y. Li, F. Javid, E. Gu, J. Jacob, S. Sueda, *et al.*, "Asap: Automated sequence planning for complex robotic assembly with physical feasibility," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 4380–4386.

[5] B. Tang, M. A. Lin, I. Akinola, A. Handa, G. S. Sukhatme, F. Ramos, D. Fox, and Y. Narang, "Industreal: Transferring contact-rich assembly tasks from simulation to reality," *arXiv preprint arXiv:2305.17110*, 2023.

[6] B. Tang, I. Akinola, J. Xu, B. Wen, A. Handa, K. Van Wyk, D. Fox, G. S. Sukhatme, F. Ramos, and Y. Narang, "Automate: Specialist and generalist assembly policies over diverse geometries," *arXiv preprint arXiv:2407.08028*, vol. 1, no. 2, 2024.

[7] Y. Tian, J. Jacob, Y. Huang, J. Zhao, E. Gu, P. Ma, A. Zhang, F. Javid, B. Romero, S. Chitta, *et al.*, "Fabrica: Dual-arm assembly of general multi-part objects via integrated planning and learning," *arXiv preprint arXiv:2506.05168*, 2025.

[8] J. Luo, Z. Hu, C. Xu, Y. L. Tan, J. Berg, A. Sharma, S. Schaal, C. Finn, A. Gupta, and S. Levine, "Serl: A software suite for sample-efficient robotic reinforcement learning," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 16 961–16 969.

[9] J. Luo, C. Xu, J. Wu, and S. Levine, "Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning," *arXiv preprint arXiv:2410.21845*, 2024.

[10] C. Xu, Q. Li, J. Luo, and S. Levine, "Rldg: Robotic generalist policy distillation via reinforcement learning," *arXiv preprint arXiv:2412.09858*, 2024.

[11] M. T. Mason, "Compliance and force control for computer controlled manipulators," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 11, no. 6, pp. 418–432, 1981.

[12] D. E. Whitney *et al.*, "Quasi-static assembly of compliantly supported rigid parts," *Journal of Dynamic Systems, Measurement, and Control*, vol. 104, no. 1, pp. 65–77, 1982.

[13] S. Dong, D. K. Jha, D. Romeres, S. Kim, D. Nikovski, and A. Rodriguez, "Tactile-rl for insertion: Generalization to objects of unknown geometry," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 6437–6443.

[14] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 23–30.

[15] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3803–3810.

[16] F. Sadeghi and S. Levine, "Cad2rl: Real single-image flight without a single real image," *arXiv preprint arXiv:1611.04201*, 2016.

[17] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. Van Wyk, A. Zhurkevich, B. Sundaralingam, *et al.*, "Dextreme: Transfer of agile in-hand manipulation from simulation to reality," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 5977–5984.

[18] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, *et al.*, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 4243–4250.

[19] S. James, P. Wohlhart, M. Kalakrishnan, D. Kalashnikov, A. Irpan, J. Ibarz, S. Levine, R. Hadsell, and K. Bousmalis, "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 12 627–12 637.

[20] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox, "Closing the sim-to-real loop: Adapting simulation randomization with real world experience," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8973–8979.

[21] X. Zhang, M. Tomizuka, and H. Li, "Bridging the sim-to-real gap with dynamic compliance tuning for industrial insertion," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 4356–4363.

[22] H. Ha, P. Florence, and S. Song, "Scaling up and distilling down: Language-guided robot skill acquisition," in *Conference on Robot Learning*. PMLR, 2023, pp. 3766–3777.

[23] J. Yamada, M. Rigter, J. Collins, and I. Posner, "Twist: Teacher-student world model distillation for efficient sim-to-real transfer," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 9190–9196.

[24] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine, "Residual reinforcement learning for robot control," *arXiv preprint arXiv:1812.03201*, 2018.

[25] ——, "Residual reinforcement learning for robot control," in *2019 international conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 6023–6029.

[26] P. Kulkarni, J. Kober, R. Babuška, and C. Della Santina, "Learning assembly tasks in a few minutes by combining impedance control and residual recurrent reinforcement learning," *Advanced Intelligent Systems*, vol. 4, no. 1, p. 2100095, 2022.

[27] X. Zhang, C. Wang, L. Sun, Z. Wu, X. Zhu, and M. Tomizuka, "Efficient sim-to-real transfer of contact-rich manipulation skills with online admittance residual learning," in *Conference on Robot Learning*. PMLR, 2023, pp. 1621–1639.

[28] L. Ankile, A. Simeonov, I. Shenfeld, M. Torne, and P. Agrawal, "From imitation to refinement–residual rl for precise assembly," *arXiv preprint arXiv:2407.16677*, 2024.

[29] Z. Zhang, Y. Wang, Z. Zhang, L. Wang, H. Huang, and Q. Cao, "A residual reinforcement learning method for robotic assembly using visual and force information," *Journal of Manufacturing Systems*, vol. 72, pp. 245–262, 2024.

[30] X. Xu, Y. Hou, C. Xin, Z. Liu, and S. Song, "Compliant residual dagger: Improving real-world contact-rich manipulation with human corrections," *arXiv preprint arXiv:2506.16685*, 2025.

[31] S. Haddadin, "The franka emika robot: A standard platform in robotics research," *IEEE Robotics & Automation Magazine*, 2024.

[32] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[33] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, *et al.*, "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," in *European conference on computer vision*. Springer, 2024, pp. 38–55.

[34] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, *et al.*, "Sam 2: Segment anything in images and videos," *arXiv preprint arXiv:2408.00714*, 2024.

[35] B. Wen, W. Yang, J. Kautz, and S. Birchfield, "Foundationpose: Unified 6d pose estimation and tracking of novel objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 17 868–17 879.

[36] P. J. Ball, L. Smith, I. Kostrikov, and S. Levine, "Efficient online reinforcement learning with offline data," in *International Conference on Machine Learning*. PMLR, 2023, pp. 1577–1594.

[37] J. Oh, Y. Guo, S. Singh, and H. Lee, "Self-imitation learning," in *International conference on machine learning*. PMLR, 2018, pp. 3878–3887.

[38] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg, "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3740–3747, 2023.

[39] K. Hu, H. Shi, Y. He, W. Wang, C. K. Liu, and S. Song, "Robot trains robot: Automatic real-world policy adaptation and learning for humanoids," *arXiv preprint arXiv:2508.12252*, 2025.

[40] Y. Guo, B. Tang, I. Akinola, D. Fox, A. Gupta, and Y. Narang, "Srsa: Skill retrieval and adaptation for robotic assembly tasks," *arXiv preprint arXiv:2503.04538*, 2025.