

Bridging Handheld and Teleoperated Supervision for Contact-Rich Manipulation via State-Gated Experts

Vidullan Surendran¹, Neehar Peri^{1,2}, David Watkins¹
RAI Institute¹, Carnegie Mellon University²

Abstract—Handheld data collection systems, such as the Universal Manipulation Interface (UMI), enable scalable data collection across diverse environments but only capture observed actions rather than the desired actions executed by a robot controller. In contrast, teleoperation captures desired actions directly, but requires a robot in-the-loop. We revisit this trade-off through the lens of *action validity* across task phases. We observe that handheld trajectories provide valid supervision in tolerant, free-space phases, but can be poor supervision targets for contact-sensitive phases, requiring high stiffness to minimize tracking errors, resulting in large contact forces. We study the interaction between these two supervision types for contact-rich manipulation and find that training policies that combine handheld data with a small number of targeted teleoperated demonstrations provide an effective hybrid strategy. Specifically, rather than teleoperating the entire task, we only collect partial teleoperated demonstrations for task segments where base handheld policies fail. However, naively mixing handheld and teleoperated phase-specific data yields worse performance than training on handheld data alone. To address this mismatch between observed and desired supervision, we propose Bi-modal Routing for Imitation Data via Gated Experts (BRIDGE), a mixture of diffusion policy experts that routes between specialist task phase heads conditioned on the current robot state. Notably, our approach enables task-phase specific use of desired actions during contact sensitive segments and improves success rates over handheld-only baselines by up to 36.7% across three contact-rich manipulation tasks. Please see our project page for accompanying videos.

I. INTRODUCTION

The Universal Manipulation Interface (UMI) [1] enables scalable, in-the-wild data collection by decoupling the collection process from specific robot hardware. However, this portability introduces an embodiment gap during on-robot policy execution, particularly for tasks involving contact forces, friction, and compliance. In contrast, teleoperation systems such as GELLO [2] and ALOHA [3] provide high-fidelity, on-robot data but sacrifice collection speed and dataset diversity, and often lack proprioceptive feedback.

In this paper, we examine the trade-off between handheld and teleoperated data through the lens of *action validity* across different task phases. To do so, we highlight the distinction between *observed* and *desired* actions. An observed action is the physically realized state of the robot end-effector (or handheld device), while a desired action is the reference trajectory sent to a robot controller. Due to finite stiffness, filtering, tracking errors, and contact interactions, observed actions often deviate from desired actions. Our key insight is that handheld data (which only captures observed actions) provides valid supervision in tolerant, free-space motion but can be inadequate in contact-sensitive phases.

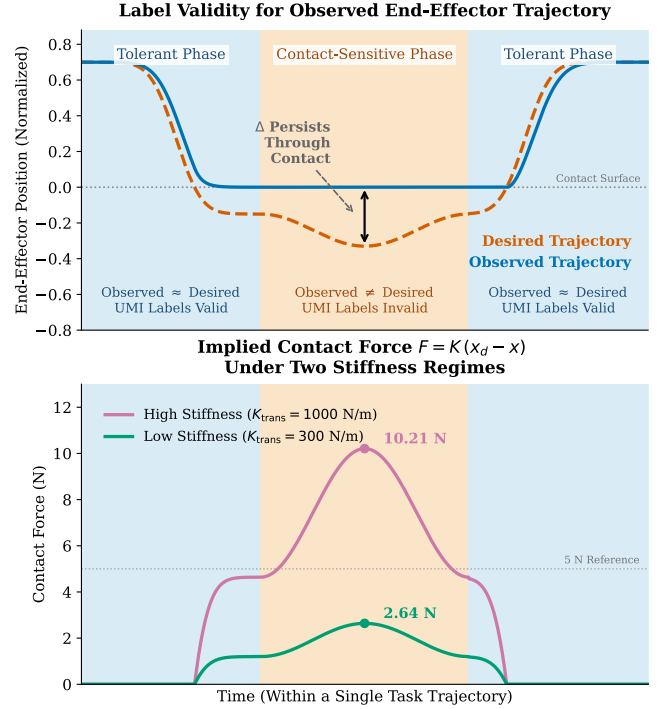


Fig. 1: Action Validity Under Contact. We illustrate the end-effector trajectory and contact forces during the NIST pulley routing task [4]. *Top:* In tolerant phases (blue), the observed action closely tracks the desired action. In contact-sensitive phases (yellow), the desired action drives below the contact surface; the resulting persistent error (Δ) makes the observed trajectory an invalid proxy for desired intent. *Bottom:* Proportional control converts Δ into contact force via $F \propto K(x_d - x)$. High stiffness gains reduce tracking error but result in large contact forces, whereas lowering stiffness reduces peak force at the cost of increased tracking error.

In these contact-rich regimes, naively tracking observed actions at high stiffness generates large contact forces (Figure 1). Unlike prior work that typically trains only on either handheld or teleoperated data, we demonstrate that augmenting handheld data with targeted teleoperated demonstrations (which capture both observed and desired actions) improves policy success during contact-sensitive task phases.

Bridging the Embodiment Gap. Despite their complementary strengths, naively combining handheld and teleoperated data often degrades policy performance because observed and desired actions do not always provide compatible supervision. To resolve this mismatch, we propose Bi-modal Routing for Imitation Data via Gated Experts (BRIDGE), a mixture of diffusion policy experts that routes between specialized task heads conditioned on the current robot state.

This framework establishes a deliberate division of labor: cheap handheld data helps learn the task scaffold, while teleoperation is reserved for contact-rich bottlenecks where observed actions fail to provide reliable supervision. We find that sparse teleoperation at contact bottlenecks approaches the performance of full-task teleoperation while reducing teleoperation data collection effort.

Contributions. We present three primary contributions. First, we study *action validity* in data collection, demonstrating that observed actions are sufficient for error-tolerant, free-space motion but are unreliable during contact-sensitive phases, where desired action supervision substantially improves execution. Second, we introduce Bi-modal Routing for Imitation Data via Gated Experts, a mixture of diffusion policy experts that jointly learns from data sources containing both observed and desired action labels. Third, we show that our approach naturally supports task-phase aware execution, allowing contact sensitive segments to use desired action supervision to improve task success rates over handheld-only baselines by up to 36.7% across three contact-rich manipulation tasks.

II. RELATED WORKS

Hybrid Data Collection Strategies. Recent work increasingly leverages data across diverse embodiments and modalities to improve policy generalization. Large-scale efforts like Open X-Embodiment [5] and DROID [6] aggregate demonstrations at scale to train foundational policies [7], [8]. Other approaches address data heterogeneity by routing through embodiment-specific heads [9], closing morphological gaps via rendering or human videos [10], and co-training across the sim-to-real boundary [11]. Alternatively, synthetic augmentation can be used to expand small demonstration sets [12]. However, naively mixing heterogeneous human demonstrations can degrade performance compared to using a single source [13]. Unlike prior work that aggregates disparate data under a shared action space, we mix two distinct types of supervision from the *same robot embodiment*. Specifically, our handheld data only provides observed actions, while the teleoperated data provides desired actions, allowing us to route between observed and desired trajectories based on local action validity.

Learning from Corrective Behaviors. Collecting targeted human demonstrations to correct residual errors in behavior cloning is an active area of research. Iterative methods like DAgger [14] query experts to label all states visited by the policy, while more recent variants reduce this burden by only querying experts in uncertain or risky states [15]. Similarly, intervention-based approaches keep a human-in-the-loop during data collection, reweighting demonstrations to emphasize corrections [16], [17]. Unlike iterative approaches, Residual Policy Learning (RPL) [18], [19] composes a learned correction over a base controller. Recent work extends RPL to contact-rich manipulation under impedance control [20]. However, these methods typically collect corrections with a model in the loop. In contrast, although our targeted

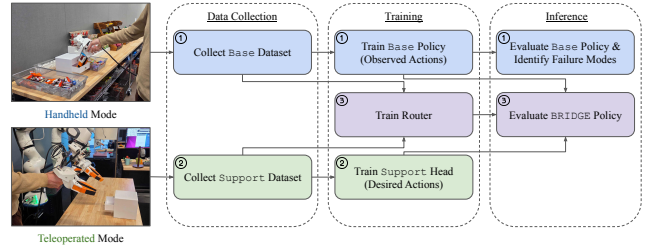


Fig. 2: **Dual Mode Data Collection Pipeline.** First, we use DM-UMI in handheld mode to collect base demonstrations to learn the task scaffold. We then train and evaluate this base policy to identify failure modes. Second, we use DM-UMI in teleoperation mode to collect a targeted support dataset to address base policy failures. We then freeze the base policy and train a support head. Third, we train a router to hard-switch between the base and support heads at inference. Our BRIDGE model combines the base policy, support head, and router (Figure 3).

demonstrations are informed by model failures, we collect our teleoperated examples in a single offline session.

Mixture-of-Experts Policies. Mixture-of-Experts (MoE) architectures, which activate sparse subnetworks per input, have been instrumental in scaling language models [21]. Recently, sparse MoE diffusion policies [22] have been adapted for contact-rich manipulation. These methods use soft routing to separate latent distributions, effectively mitigating mode averaging across heterogeneous demonstrations. Although our architecture shares the high-level intuition of routing inputs to specialized experts, we neither scale our parameter count nor train the router implicitly using a standard task loss. Instead, our experts correspond to two distinct sources of supervision.

III. LEARNING A STATE-GATED MIXTURE OF EXPERTS

In this section, we present DM-UMI, our dual-mode data collection setup for contact-rich manipulation. We then introduce BRIDGE, our mixture-of-experts policy architecture that learns from heterogeneous supervision. Unlike residual policies, BRIDGE does not require the support expert to learn a correction over the full base action distribution. Instead, it treats observed and desired actions as distinct supervision types and routes to the support expert when in the support action manifold.

DM-UMI: Dual-Mode Data Collection Device. UMI [1] relies on offline monocular SLAM for trajectory reconstruction and ArUco tag detection for gripper width estimation. While this enables in-the-wild data collection, it requires offline processing to extract camera pose and gripper width. Since teleoperation requires online estimates of these parameters, we design a dual-mode handheld device called Dual-Mode UMI (DM-UMI) that modifies UMI’s sensing stack to support both handheld and teleoperated data collection. DM-UMI uses an XVISIO DS80 SLAM camera to provide online 6-DoF pose estimates, and magnetic position sensors to measure gripper width. We use these pose and width values to command the robot end effector and gripper.

Dataset Collection. We use DM-UMI to collect both handheld and teleoperated demonstrations (Figure 2). First,

we use DM-UMI to collect task demonstrations across diverse environments, referred to as the `base` dataset. We train a `Base` policy on this data and empirically identify task phases associated with systematic failures. Next, we teleoperate the robot using the DM-UMI to collect targeted demonstrations for these phases in a single pass rather than through iterative policy-in-the-loop data collection. This support dataset implicitly captures embodiment-specific effects, including controller dynamics, robot kinematics, contact dynamics, and grasp stability. This approach leverages the complementary structure of both datasets, where the `base` dataset (observed actions only) provides nominal task coverage, while the support dataset (desired actions) isolates localized regimes where the `base` policy is unreliable. We describe the `base` policy, and present our approach below (Figure 3).

Stage 1: Base Policy Training. Our base policy architecture is derived from Diffusion Policy [23]. Similar to prior work, we model action prediction as denoising future action chunks conditioned on visual and proprioceptive observations. However, we introduce several architectural modifications motivated by the demands of precise manipulation, multiple data streams, and heterogeneous supervision. Many diffusion-based policies [23], [1] extract visual features from a ViT encoder and only use the class (CLS) token as a global visual descriptor. While efficient, this compresses spatial structure early in the model, making it difficult to reason about the spatial relationship between the end effector and the environment for precise, contact-rich tasks. Therefore, we retain the full set of spatial patch tokens produced by the DINOv2 [24] vision encoder. For a batch size B , a backbone with patch size N , and a latent dimension D , an image is encoded into $V \in \mathbb{R}^{B \times N \times D}$ tokens. Given a 518×518 image and a patch size of 14, DINOv2 produces 1369 spatial tokens. Therefore, we employ a PerceiverIO-style [25] aggregation to reduce the number of tokens with a set of learnable queries, $Q \in \mathbb{R}^{B \times M \times D}$, $M \ll N$ via stacked cross-attention blocks $Z_{vision} = XATTN(Q, V)$. This information bottleneck effectively extracts task-relevant structure while discarding irrelevant visual details. Similarly, state inputs such as end-effector pose and gripper width are concatenated and linearly projected into $\mathbb{R}^D$ as Z_{state} and are cross-attended with the vision tokens to generate the latent conditioning $Z_{latent} = XATTN(Z_{state}, Z_{vision})$. We refer to the modules that generate Z_{vision} , Z_{state} and Z_{latent} as the latent adapter ϕ_b . We follow the standard conditional denoising formulation used in Chi et al. [23] where Z_{latent} is used as input for a temporal diffusion policy, and train with standard diffusion losses on the `base` dataset.

Stage 2: Support Head Training. With the `base` expert π_b and shared vision encoder kept frozen, we train the support latent adapter ϕ_s , and head π_s on the support dataset. The action target for this expert is the desired trajectory, allowing it to map the latent to a desired action chunk, $\hat{a}_{t:t+H}^s = \pi_s(\phi_s(Z_{latent}))$. The support expert is optimized independently with the standard diffusion loss.

Stage 3: Router Training. Finally, we train a MLP gate

$G_\psi(z)$ to dynamically route observations to the appropriate expert. To train this gate, we extract the intermediate Z_{latent} from π_b for samples in both the `base` and support datasets. To avoid ambiguous pseudo-labels in regions where `base` and support demonstrations overlap, we remove `base` latents within an ϵ -neighborhood of support latents before constructing the router training set. Given the filtered `base` latents B_b and support latent bank B_s , we compute the similarity to the $k = 16$ neighbors in each bank,

$$\begin{aligned}\sigma_+(z) &= \frac{1}{k} \sum_{z_s \in \mathcal{N}_k(z, B_s)} \cos(z, z_s), \\ \sigma_-(z) &= \frac{1}{k} \sum_{z_b \in \mathcal{N}_k(z, B_b)} \cos(z, z_b)\end{aligned}\quad (1)$$

To account for overlap between the local neighborhoods, we use a class relative score with a decision threshold η_ρ to generate the label $\rho(z)$,

$$\rho(z) = (\sigma_+(z) - \sigma_-(z)) > \eta_\rho, \quad (2)$$

Lastly, we distill this k -NN classifier into G_ψ using a binary cross-entropy loss, and train it to predict $\rho(z)$.

Router Architecture. The router follows the same structure as the `base` policy with the action head replaced by a lightweight residual MLP. We reuse the DINOv2 tokens and the state input from the `base` policy. The state inputs are linearly projected to the latent dimension of $d = 384$, and fused with the vision tokens using a PerceiverIO-style aggregation. The output tokens are layer normalized and passed through a two-layer MLP with GELU activation. The final linear layer of the adapter is zero-initialized, so that the adapter initially acts as an identity mapping.

BRIDGE Inference. During inference, each observation o at time t is encoded into shared visual and state features. The `base` and support adapters produce expert-specific latents $z_t^b = \phi_b(Z_{vision}, Z_{state})$ and $z_t^s = \phi_s(Z_{vision}, Z_{state})$. The router predicts support probabilities $g_{t:t+H} = G_\psi(z_t^b)$. Given an inference threshold η , we hard switch via,

$$\hat{a}_{t+j}^b = \pi_b(z_t^b), \quad \hat{a}_{t+j}^s = \pi_s(z_t^s) \quad (3)$$

$$m_{t+j} = \mathbb{I}[g_{t+j} > \eta], \quad \hat{a}_{t+j} = (1 - m_{t+j})\hat{a}_{t+j}^b + m_{t+j}\hat{a}_{t+j}^s \quad (4)$$

Why Use Hard-Switching for Routing? During inference, the router $G_\psi(z_t)$ selects between the `base` and support experts using a hard switch. We use hard switching rather than action blending because the two experts may represent distinct behavioral modes. The `base` policy is trained on observed actions, while the support expert is trained on desired actions, as discussed in Section I. Averaging actions from these experts can therefore produce actions that are not meaningful for either mode. Although hard switching can introduce boundary sensitivity, the downstream control stack smooths switching discontinuities. The selected policy output is passed to a trajectory generator that interpolates commanded motion at 100 Hz, which is then streamed to a 1 kHz Cartesian impedance controller with torque smoothing. During rollouts, we damp transient

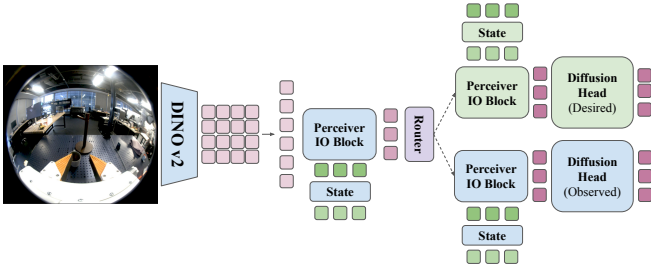


Fig. 3: **Model Architecture.** We propose BRIDGE, an extension of Diffusion Policy that dynamically routes between predicting observed and desired actions. Visual observations are encoded via DINOv2 [24], processed through a Perceiver IO block, and fused with state features via cross-attention. This shared latent representation is passed to a router, which switches between the observed and desired diffusion heads. To train BRIDGE, we sequentially train the *base* model (shown in blue), collect targeted teleoperated demonstrations to train the *support* head (green), and finally train the router (purple).

switching by adding temporal hysteresis that activates the support expert only when $G_\psi(z_t) > \eta$, with $\eta = 0.5$, for a continuous duration of 0.25 seconds.

IV. EXPERIMENTS

We evaluate baseline policies on three contact-rich manipulation tasks and demonstrate that BRIDGE consistently outperforms handheld baselines and approaches the performance of policies trained on full teleoperated data. Lastly, we characterize the performance of our router.

Dual-Mode System Characterization. We evaluate the DS80’s pose tracking accuracy in Table I. To characterize the behavior of our dual-mode data collection system, we collected 1000 DM-UMI demonstrations in a motion capture environment (20 Vicon Vantage V16 cameras). The system exhibits a mean absolute translation error of 25.9 mm and a mean absolute rotation error of 6.3 degrees. However, its precision improves substantially when measuring relative poses. The mean relative translation error decreases to 1.1 mm, while the relative rotation error drops to 0.2 degrees, which is a more relevant measure for our policies, since they are trained to predict relative poses.

We find that handheld DM-UMI demonstrations are faster to collect than teleoperated ones (Table II). Across all evaluated tasks, handheld collection is roughly $3\times$ faster than full-task teleoperation, and partial teleoperation is roughly

TABLE I: **DS80 Pose Tracking Accuracy.** We collect a dataset of 1000 DM-UMI demonstrations interacting with diverse objects in a motion capture room to precisely evaluate the DS80’s absolute and relative 6-DoF pose error. Across all demonstrations, we find that the DS80 has an average absolute translation error of 25.9 mm and an absolute rotation error of 6.3 degrees. However, since we train our policies with relative poses, tracking error drops significantly.

	Abs. Translation (mm)	Rel. Translation (mm)	Abs. Rotation (deg)	Rel. Rotation (deg)
Median	22.6	0.9	4.0	0.2
Mean	25.9	1.1	6.3	0.2
STD	16.1	0.7	5.1	0.2

TABLE II: **Comparison of Episode Duration.** We present a comparison of the average time of a single demonstration (seconds $\pm$ std-dev) with our DM-UMI device for handheld, targeted teleoperation, and full-task teleoperation respectively. Full task teleoperated demonstrations were collected under a comparable data collection time budget, which included task resets and collector breaks.

Task	Handheld	Targeted Teleoperation	Full-Task Teleoperation
Pipe Insertion	6.7 ± 1.6	6.7 ± 2.0	15.3 ± 2.2
Battery Insertion	11.1 ± 1.7	15.5 ± 3.6	27.6 ± 4.0
NIST Pulley Task	10.5 ± 2.6	9.7 ± 4.5	24.7 ± 3.3

$2\times$ faster. However, these runtime statistics still underestimate the practical cost of full teleoperation, which is more physically tiring. This cost is also highly task-dependent. For short, spatially compact tasks like pipe insertion, experienced operators collect full-task demonstrations efficiently. However, this efficiency degrades in longer-horizon tasks requiring large workspace motions and sustained contact. In pulley routing, for example, the operator must maintain O-ring tension over a wide range of motion without direct force feedback. We found this substantially more difficult; not only was the operator forced into awkward postures, but the robot safety limit faults were triggered frequently during collection. In contrast, BRIDGE decouples broad nominal coverage from localized teleoperation. Handheld demonstrations provide the foundational task scaffold, while teleoperated *support* is collected exclusively around failure regions where observed action labels are inadequate. Therefore, even when full teleoperation provides a strong in-domain upper bound, targeted *support* collection minimizes the physical burden of continuous teleoperation and eliminates the redundancy of repeatedly teleoperating trivial free-space phases.

Dataset. We summarize the number of demonstrations collected per task in Table III. While *support* data often occupies a large fraction of the trajectories’ temporal extent due to slower collection rates, it remains spatially sparse. We collect handheld demonstrations for each task until the *base* policy converges during the free-space task phases. Rather than relying on exhaustive data collection, we demonstrate that targeted, small-scale *support* demonstrations are sufficient to significantly improve task success. Collecting the full-task teleoperated dataset was considerably more cognitively and physically demanding than collecting the *base-plus-support* dataset.

Tasks. We evaluate our method on three tasks (Figure 4) that require precise geometric alignment and stable contact handling under external forces. These tasks highlight the need for distinguishing *observed* and *desired* actions:

- *NIST Pulley Routing.* The robot must grasp a deformable O-ring and route it around small and large pulleys. Adapted from NIST Assembly Task Board #2 [4], this requires maintaining constant O-ring tension during precise end-effector movements.
- *Pipe Insertion.* The robot must grasp a pipe and insert it into a tight-tolerance opening. This requires reliable

TABLE III: **Dataset Statistics.** We summarize the number of collected base, support, and fully teleoperated demonstrations below. Support Temporal % is the median fraction of time and support distance % is the median fraction of the end-effector path length covered by a support demonstration.

Task	Base #	Support #	Teleop #	Support Temporal %	Support Dist. %
NIST Pulley Routing	201	50	60	37.8%	7.4%
Pipe Insertion	100	81	60	24.5%	3.4%
Battery Insertion	290	59	100	61.1%	14.9%

TABLE IV: **NIST Pulley Routing.** We report the results for 5 trials in each of the (C)enter, (L)eft, (R)ight, (T)op, and (B)ottom positions. We shift the rig ± 2 cm in each direction.

Method	Sub-task Performance (C-L-R-T-B)		
	Full (1.0)	Score	Task Success Rate
Base Policy (Handheld)	4-5-2-0-0	11/25	44.0%
Naive Mix	0-0-0-0-0	0/25	0.0%
BRIDGE (Ours)	4-4-4-4-3	19/25	76.0%
Base Policy (Teleoperated)	5-5-4-4-3	21/25	84.0%

grasping, 6-DoF alignment, and force-aware insertion.

- **Battery Insertion.** The robot picks up a AA battery and inserts it into a spring-loaded compartment. This tests end-effector control under external forces, as the spring must be kept compressed while the battery is seated.

Baselines. We evaluated the policies on a Franka FR3 7-DoF arm running a Cartesian impedance controller. All evaluated methods are based on the architecture described in Section III, differing only in their supervision strategy and training schedule. Similar to Chi et al. [1], we used the relative end-effector and absolute gripper width formulation, and an action horizon of 24 steps.

- **Base Policy (Handheld).** We train exclusively on handheld DM-UMI demonstrations (observed actions).
- **Base Policy (Teleoperated).** We train exclusively on teleoperated demonstrations (desired actions) as an upper bound.
- **Naive Mix.** We train the Base policy on a uniformly sampled combination of handheld and targeted teleoperated demonstrations.
- **BRIDGE (Ours).** We train the BRIDGE model on the base and support datasets following the procedure described in Section III.

NIST Pulley Routing Results. This task evaluates spatial robustness during loaded routing, where the robot must maintain O-ring tension while moving around the pulley fixture. As shown in Table IV, the Base Policy (Handheld) succeeds in 44.0% of trials, but its performance drops sharply in the top and bottom fixture positions. The Naive Mix baseline fails because mixing observed and desired labels leads to imprecise manipulation; in practice, the policy often touches the top of the pulley and misses the routing groove. In contrast, BRIDGE routes to the support expert for loaded routing and executes this segment with lower stiffness, improving success to 76.0%, approaching the full teleoperated policy at 84.0%. These results suggest that targeted support data can recover much of the benefit of teleoperation while avoiding the need to teleoperate the full task.

TABLE V: **Pipe Insertion Evaluation.** We report the results for 10 trials in each of the (L)eft, (C)enter, and (R)ight positions. We shift the rig ± 2 cm in each direction.

Method (Points)	Sub-task Performance (L-C-R)				Score	Task Success Rate
	Full (1.0)	Partial Insert (0.75)	$\geq 50\%$ Aligned Not Inserted (0.5)	Failed Alignment (0.0)		
Base Policy (Handheld)	0-2-2	0-1-0	10-7-8	0-0-0	17.25/30	13.3%
Naive Mix	0-0-2	0-0-4	0-4-2	10-6-2	8/30	6.7%
BRIDGE (Ours)	3-7-5	1-2-1	4-0-4	2-1-0	22.00/30	50.0%
Base Policy (Teleoperated)	3-7-9	1-1-1	5-0-0	1-2-0	23.75/30	63.3%

TABLE VI: **Spring-Loaded Battery Evaluation.** We report the results for 10 trials in each of the (L)eft, (C)enter, and (R)ight positions. We shift the rig ± 1 cm in each direction.

Method (Points)	Sub-task Performance (L-C-R)					Score	Task Success Rate
	Full (1.0)	Not Fully Inserted (0.75)	Improperly Placed (0.5)	Ejected or Dropped (0.25)	Missed Pick (0.0)		
Base Policy (Handheld)	0-3-0	1-0-1	7-1-2	1-2-4	1-4-3	11.25/30	10.0%
Naive Mix	0-0-0	0-0-0	0-0-0	3-4-4	7-6-6	2.75/30	0.0%
BRIDGE (Ours)	3-4-3	4-4-1	1-1-2	1-1-2	1-0-2	19.75/30	33.3%
Base Policy (Teleoperated)	2-7-3	3-1-5	1-0-0	2-0-0	2-2-2	19.75/30	40.0%

Pipe Insertion Results. This task tests geometric precision as small pose errors easily cause jamming. As shown in Table V, the Base Policy (Handheld) frequently aligns the pipe but fails at insertion due to tracking errors near contact (13.3% success). Naive Mix performs far below the base handheld policy. This suggests that even with high stiffness, combining observed and desired action spaces into a single policy produces inconsistent alignment and contact behavior. High stiffness can partially compensate for the mismatch in some cases, but it does not replace the need for separating the observed and desired action spaces. In contrast, BRIDGE uses targeted switching to command desired actions during the final alignment and insertion phase. This quadruples the success rate to 50.0%, substantially closing the gap toward the teleoperated upper bound (63.3%).

Spring-Loaded Battery Insertion Results. This task demonstrates that action mismatch is primarily localized to contact phases (Table VI). Although the Naive Mix picks up the battery roughly 40% of the time, it exhibits a systematic lateral bias that shifts the battery rightward during transport, resulting in an insertion failure. This suggests that directly mixing observed and desired labels can distort the learned action distribution even before the final contact phase. The Base Policy (Handheld) manages free-space transport well but struggles against the spring terminal, yielding a 10.0% success rate. By preserving this free-space competence while using support demonstrations to correct the delicate insertion phase, BRIDGE more than triples the success rate to 33.3%. Notably, BRIDGE boosts the partial credit score from 11.25 to 19.75 and more unsuccessful trials get closer to completion (e.g., reaching the “Not Fully Inserted” stage).

Statistical Analysis. We compute Wilson 95% confidence

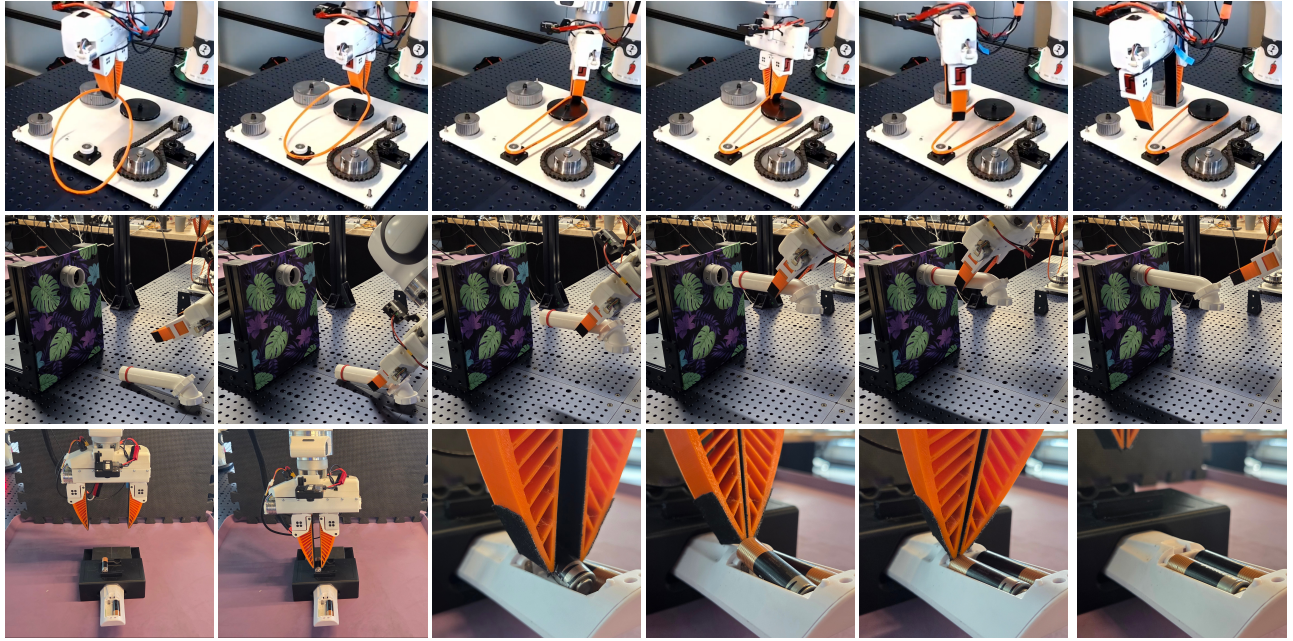


Fig. 4: **Policy Rollouts.** We evaluate three precise, contact-rich tasks, including NIST pulley routing (*top*), pipe insertion (*middle*), and spring-loaded battery insertion (*bottom*).

intervals over rollout success and two-sided Fisher exact tests comparing BRIDGE against the handheld and full-teleoperation baselines. BRIDGE improves over handheld supervision on pulley routing (76.0% vs. 44.0%, $p = 0.042$) and pipe insertion (50.0% vs. 13.3%, $p = 0.005$); the battery improvement (33.3% vs. 10.0%) does not reach $p < 0.05$ at the evaluated trial count with $p = 0.057$. BRIDGE is not statistically separable from full teleoperation on pulley, pipe, or battery ($p = 0.73, 0.44, 0.79$, respectively). These intervals quantify rollout uncertainty rather than variation across independently trained policy seeds.

Router Analysis. We evaluate our router on the challenging pipe insertion task. We use a held-out teleoperated dataset with manually labeled support phases to prevent the router evaluation from artificially inflating performance by exploiting dataset or embodiment artifacts. On this dataset, our MLP router achieves 99.0% recall and 69.0% precision. High recall is desirable to prevent the base policy from incorrectly handling insertion alignment. However, because the transition between base and support phases is inherently ambiguous, frame-level labels are imprecise near the boundary. Figure 5 shows that false positives concentrate near the support manifold boundary (representing transition states) rather than unrelated base states. This aligns with our nearest-neighbor design, where the support bank covers the support-state distribution, naturally restricting successful routing to this manifold rather than requiring extrapolation to disjoint regions.

Preliminary Exploration of Data Scaling. We investigate the impact of handheld and targeted teleoperated demonstrations on the pipe insertion task. Table VII summarizes these initial scaling results. Training a base expert on 120 fully teleoperated demonstrations yields a 93.3% success rate, significantly improving over the 63.3% achieved by the 60

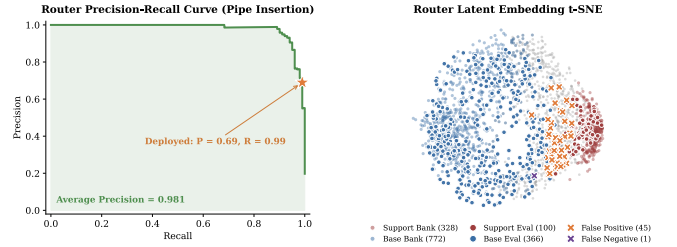


Fig. 5: **Router Analysis.** We visualize the precision-recall curve for the pipe insertion task (left). The deployed router achieves 99.0% recall and 69.0% precision, favoring early support activation over missed handoffs. Computed t-SNE latent embeddings demonstrate clear separation between base and support states, yielding only a single false negative (right).

TABLE VII: **Pipe Insertion Scaling Ablations.** We report the results for 10 trials in each of the (L)eft, (C)enter, and (R)ight positions. We shift the rig ± 2 cm in each direction.

Method (Points)	Sub-task Performance (L-C-R)				Score	Task Success Rate
	Full (1.0)	Partial Insert (0.75)	$\geq 50\%$ Aligned Not Inserted (0.5)	Failed Alignment (0.0)		
Base Policy (Full Teleop, 60 demos)	3-7-9	1-1-1	5-0-0	1-2-0	23.75/30	63.3%
Base Policy (Full Teleop, 120 demos)	9-10-9	0-0-0	1-0-1	0-0-0	29.00/30	93.3%
Base Policy (Handheld, 100 demos)	0-2-2	0-1-0	10-7-8	0-0-0	17.25/30	13.3%
Base Policy (Handheld, 200 demos)	0-4-3	0-0-0	10-6-7	0-0-0	18.50/30	23.3%
Base Policy (Handheld, 328 demos)	0-5-3	0-0-0	10-5-7	0-0-0	19.00/30	26.7%
BRIDGE (100 Handheld + 81 Partial Teleop)	3-7-5	1-2-1	4-0-4	2-1-0	22.00/30	50.0%
BRIDGE (100 Handheld + 200 Partial Teleop)	6-8-8	2-2-0	2-0-2	0-0-0	27.00/30	73.3%

demonstration baseline. This high performance is expected; teleoperated data is collected directly on the evaluation robot and pipe insertion setup, effectively minimizing the embodiment gap.

Increasing the number of handheld demonstrations alone provides marginal gains. Doubling the original 100 demonstration dataset to 200 demonstrations achieves a 23.3% full-insertion success rate and a score of 18.50/30. Scaling further to 328 UMI demonstrations slightly increases success to 26.7% and the score to 19.00/30. These results suggest that the primary failure mode is not insufficient coverage of nom-

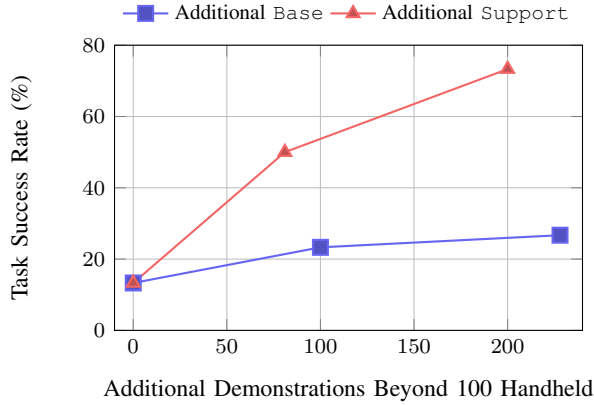


Fig. 6: **Impact of Scaling Targeted Support Data.** Starting from the same 100-demo base policy, incorporating targeted partial teleoperation demonstrations improves pipe insertion success substantially more than simply adding more handheld demonstrations. While adding over 200 additional handheld demonstrations yields only marginal improvements, utilizing our targeted partial teleoperation approach increases task success to over 70%.

inal UMI behavior, but rather imperfect trajectory tracking under contact-rich regimes. In contrast, scaling the partial teleoperation support data from 81 to 200 demonstrations yields substantial improvements. By combining the original 100 demonstration handheld base expert with a support expert trained on this expanded dataset, BRIDGE achieves a 73.3% full-insertion success rate and a score of 27.00/30. This is a marked improvement over the 50.0% success and 22.00/30 score achieved with the original 81 demonstration support set. This improvement stems from converting “aligned-but-not-inserted” states into full insertions.

Cross-Robot Teleoperation Transfer. The 120 demonstration full teleoperation pipe insertion policy with a success rate of 93.3% represents an in-domain upper bound. To examine hardware sensitivity, we evaluated it on a second Franka FR3 (Robot B) for 10 trials with the pipe at the Center position. Full task success rate dropped sharply to 10%, with most trials reaching only partial alignment. This outcome aligns with the known sensitivity of policies to variations in robot calibration, controller dynamics, and minor setup biases. We then augmented the handheld base policy with 50 targeted support demonstrations collected on Robot B in about 15 minutes wall time. BRIDGE achieved a full task success rate of 30% with a task completion score of 7.00/10 (3 full, 4 partial, 2 aligned, 1 failure). While further investigation is required, this preliminary result suggests that a reusable handheld task scaffold can help recover cross-robot performance through targeted teleoperation, reducing reliance on full task teleoperation.

Ablation on Compliance and Support Supervision. We ablate BRIDGE’s controller stiffness, and action supervision used to train the support head on the pulley task (Table VIII). Lowering the controller stiffness to LOW and using the Base Policy (Handheld) drops success from 44.0% to 24.0%, and scheduling its impedance with BRIDGE’s router

TABLE VIII: **Ablation on Compliance and Support Supervision.** We ablate the impedance schedule and support-head supervision on the pulley task below. Lowering the Base Policy (Handheld) from HIGH to LOW stiffness reduces success from 44% to 24%, and scheduling stiffness with BRIDGE’s router (State-Gated Stiffness) recovers only 28%, compared to BRIDGE’s 76%. Training the support head on observed rather than desired actions (OBS Support) reaches 20%. Importantly, this suggests that BRIDGE’s gain comes from routing between policies rather than from compliance alone. We denote the Base policy with B and the Support policy with S, where HIGH uses 1600 N/m and 60 Nm/rad and LOW uses 800 N/m and 30 Nm/rad, both critically damped. All other controller parameters are identical across methods.

Method	Compliance	Task Success Rate
Teleoperation	LOW	21/25 (84%)
BRIDGE	B:HIGH, S:LOW	19/25 (76%)
Handheld	HIGH	11/25 (44%)
Handheld	LOW	6/25 (24%)
Handheld, State-Gated Stiffness	HIGH/LOW	7/25 (28%)
BRIDGE, OBS Support	B:HIGH, S:LOW	5/25 (20%)

(Handheld, State-Gated Stiffness) reaches only 28.0%, compared to BRIDGE’s 76.0%. This suggests that the change in controller stiffness when switching to the support head is not the sole reason for improvement. We find that the source of action supervision matters as well. We retrain the support head on observed rather than desired actions (BRIDGE, OBS Support), using the same teleoperated support trajectories with all else identical. This variant achieves 5/25 (20.0%) compared to BRIDGE’s 19/25 (76.0%).

Ablation on Router Labels. Training the task router on raw source labels rather than kNN pseudo-labels achieves 100% recall and 62.8% precision ($\eta=0.5$) on the held-out test set. However, this router succeeds on only 9/25 (36%) trials on the robot, compared to 19/25 (76%) for our kNN pseudo-label router and 22/25 (88%) for an oracle router that we construct by manually switching between policies, suggesting that ϵ -filtering the kNN pseudo-labels to remove overlap is critical.

Visualization of Support Regions. Figure 7 illustrates the 3D end-effector trajectories for battery insertion, pipe insertion, and NIST pulley routing. Each plot displays a single episode’s full trajectory, originating at the start position (marked by a black dot) and concluding at the end position (marked by a yellow star). Within each blue trajectory path, the critical phase (identified as the support segment) is highlighted in red. Specifically, the median support segment accounts for 14.9% of the total trajectory length in the battery insertion task, 3.4% in the pipe insertion task, and 7.4% in the NIST pulley routing task.

Limitations and Future Work. Our approach for identifying which task phases require support data requires training and evaluating a base policy. Future work should formalize this criterion so that it can be estimated automati-

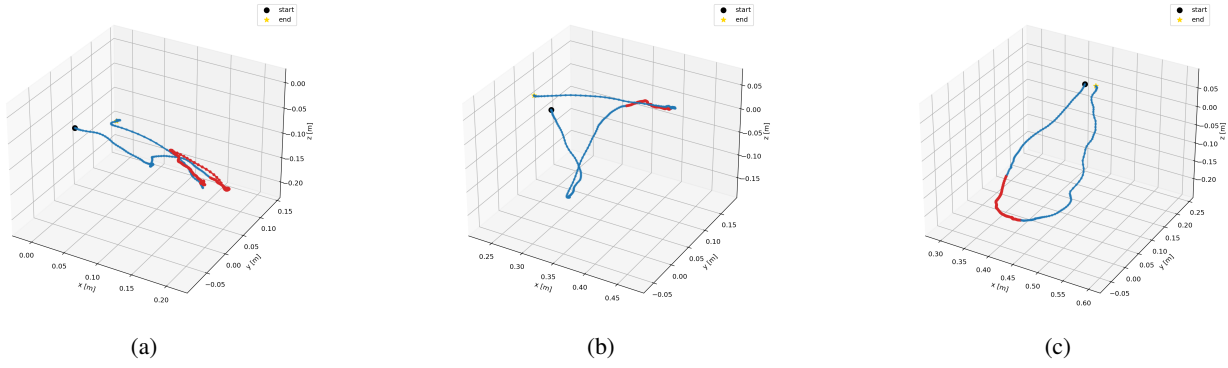


Fig. 7: **Trajectory Support Coverage Across Tasks.** As illustrated by the red highlighted segments, the median trajectory support coverage spans 14.9% of the trajectory length for battery insertion (a), 3.4% for pipe insertion (b), and 7.4% for NIST pulley routing (c).

cally from the training data. Rather than inferring support regions from a trained base model, UMI-style devices instrumented with force-torque sensing [26] may allow us to identify them directly.

V. CONCLUSION

In this work, we address the embodiment gap between handheld data collection and teleoperation through the lens of *action validity*. We demonstrate that handheld data, which provides observed actions, offers valid supervision for error-tolerant phases but can fail during contact-sensitive phases that require dynamically feasible desired actions. To improve data collection efficiency, we propose Bi-modal Routing for Imitation Data via Gated Experts (BRIDGE), a mixture of diffusion policy experts that uses scalable handheld data to learn the general task scaffold and sparse teleoperated data to target contact-rich bottlenecks. Across our evaluated tasks, this sparse teleoperated data covers less than 15% of the end-effector path length, yet BRIDGE achieves at least 79% of the full teleoperation upper bound success rate. By separating observed and desired action heads, BRIDGE provides a practical mechanism for specializing behavior across task phases.

REFERENCES

- [1] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song, “Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots,” *arXiv:2402.10329*, 2024.
- [2] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel, “Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators,” *arXiv:2309.13037*, 2023.
- [3] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, “Learning fine-grained bimanual manipulation with low-cost hardware,” *arXiv:2304.13705*, 2023.
- [4] NIST, “Assembly performance metrics and test methods,” 2026.
- [5] O. X.-E. Collaboration, “Open x-embodiment: Robotic learning datasets and rt-x models,” *arXiv:2310.08864*, 2023.
- [6] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, Y. Zhu, C. Finn, S. Levine, and P. Liang, “Droid: A large-scale in-the-wild robot manipulation dataset,” *arXiv:2403.12945*, 2024.
- [7] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, *et al.*, “Octo: An open-source generalist robot policy,” *arXiv:2405.12213*, 2024.
- [8] P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, *et al.*, “Pi0.5: a vision-language-action model with open-world generalization,” *arXiv:2504.16054*, 2025.
- [9] L. Wang, X. Chen, J. Zhao, and K. He, “Scaling proprioceptive-visual learning with heterogeneous pre-trained transformers,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.
- [10] Z. Fu, Q. Zhao, Q. Wu, G. Wetzstein, and C. Finn, “HumanPlus: Humanoid shadowing and imitation from humans,” in *CoRL*, 2024.
- [11] M. Torne, A. Simeonov, Z. Li, A. Chan, T. Chen, A. Gupta, and P. Agrawal, “Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation,” in *RSS*, 2024.
- [12] A. Mandelkar, S. Nasiriany, B. Wen, I. Akinola, Y. Narang, L. Fan, Y. Zhu, and D. Fox, “Mimicgen: A data generation system for scalable robot learning using human demonstrations,” in *CoRL*, 2023.
- [13] A. Mandelkar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, “What matters in learning from offline human demonstrations for robot manipulation,” in *CoRL*, 2021.
- [14] S. Ross, G. Gordon, and D. Bagnell, “A reduction of imitation learning and structured prediction to no-regret online learning,” in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.
- [15] R. Hoque, A. Balakrishna, E. Novoseller, A. Wilcox, D. S. Brown, and K. Goldberg, “ThriftyDagger: Budget-aware novelty and risk gating for interactive imitation learning,” in *CoRL*, 2021.
- [16] J. Spencer, S. Choudhury, M. Barnes, M. Schmittle, M. Chiang, P. Ramadge, and S. Srinivasa, “Learning from interventions: Human-robot interaction as both explicit and implicit feedback,” in *RSS*, 2020.
- [17] H. Liu, S. Nasiriany, L. Zhang, Z. Bao, and Y. Zhu, “Robot learning on the job: Human-in-the-loop autonomy and learning during deployment,” in *RSS*, 2023.
- [18] T. Silver, K. Allen, J. Tenenbaum, and L. Kaelbling, “Residual policy learning,” *arXiv:1812.06298*, 2018.
- [19] 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,” in *ICRA*, 2019.
- [20] X. Xu, Y. Hou, C. Xin, Z. Liu, and S. Song, “Compliant residual DAgger: Improving real-world contact-rich manipulation with human corrections,” *arXiv:2506.16685*, 2025.
- [21] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, “Outrageously large neural networks: The sparsely-gated mixture-of-experts layer,” *arXiv:1701.06538*, 2017.
- [22] C. Hao, X. Zhai, Y. Liu, and H. Soh, “Abstracting robot manipulation skills via mixture-of-experts diffusion policies,” *arXiv:2601.21251*, 2026.
- [23] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, “Diffusion policy: Visuomotor policy learning via action diffusion,” *IJRR*, 2025.
- [24] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, *et al.*, “Dinov2: Learning robust visual features without supervision,” *arXiv:2304.07193*, 2023.
- [25] A. Jaegle, S. Borgeaud, J.-B. Alayrac, C. Doersch, C. Ionescu, D. Ding, S. Koppula, D. Zoran, A. Brock, E. Shelhamer, *et al.*, “Perceiver io: A general architecture for structured inputs & outputs,” *arXiv:2107.14795*, 2021.
- [26] H. Choi, Y. Hou, C. Pan, S. Hong, A. Patel, X. Xu, M. R. Cutkosky, and S. Song, “In-the-wild compliant manipulation with umi-ft,” *arXiv:2601.09988*, 2026.