

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

Vidullan Surendran¹, Neehar Peri^{1,2}, David Watkins¹

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 is prohibitively time-consuming to collect. 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 lack dynamic feasibility in contact-sensitive phases, where tracking observed trajectories at high stiffness produces large, unsafe 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 efficient 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. See our [project page](#) for videos.

Keywords: Mixture-of-Experts Diffusion Policy, Universal Manipulation Interface, Heterogeneous Data Collection, Contact-Rich Manipulation

1 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 mechanical 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 lacks dynamic feasibility in contact-sensitive phases. In these contact-rich regimes, naively tracking observed actions at high stiffness generates large, unsafe contact forces (Figure 1, illustrative; see supplement for measured contact-force traces). Unlike prior work that typically trains only on either handheld or teleoperated data,

¹ RAI Institute, ² Carnegie Mellon University

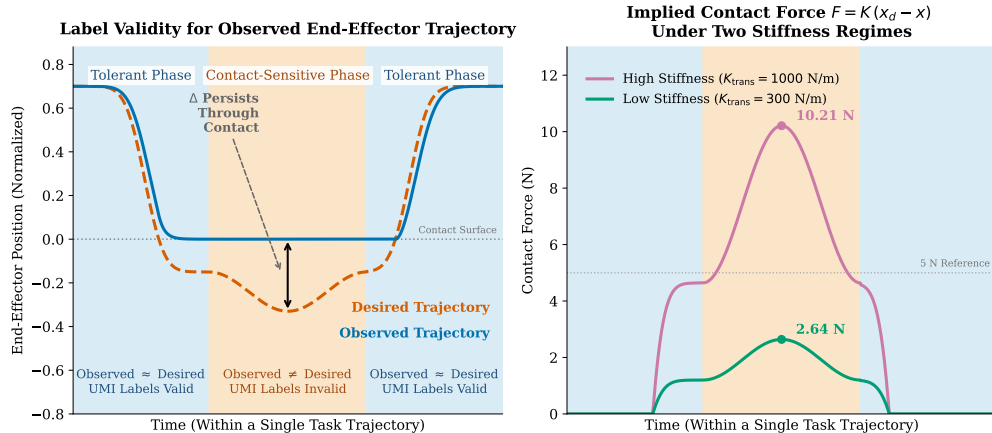


Figure 1: **Action Validity Under Contact (Illustrative)**. We visualize the end-effector trajectory and contact forces during the NIST pulley routing task [4]; real data is provided in the supplement. *Left*: In tolerant phases (blue), the observed action closely approximates the desired action. In 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. *Right*: Proportional control converts Δ into contact force via $F \propto K(x_d - x)$. High stiffness gain tracks the observed trajectory closely but at the cost of larger peak contact forces; low stiffness reduces peak force but cannot track observed trajectories through contact.

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 without significantly increasing data-collection effort.

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. This localized approach yields two key benefits. First, we find that sparse teleoperation at contact bottlenecks approaches the performance of full-task teleoperation while substantially reducing data collection effort. Second, we note that this same gating logic can also be used to select phase appropriate controller settings, though we primarily focus on routing between sources of action supervision in this paper.

Contributions. We present three primary contributions. First, we formalize *action validity* in data collection, demonstrating that observed actions are sufficient for error-tolerant, free-space motion but fail during force-sensitive phases, where desired actions are necessary for precise and compliant 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.

2 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, 10], closing morphological gaps via rendering [11] or human videos [12], and co-training across the sim-to-real boundary [13]. Alternatively, synthetic augmentation can be used to expand

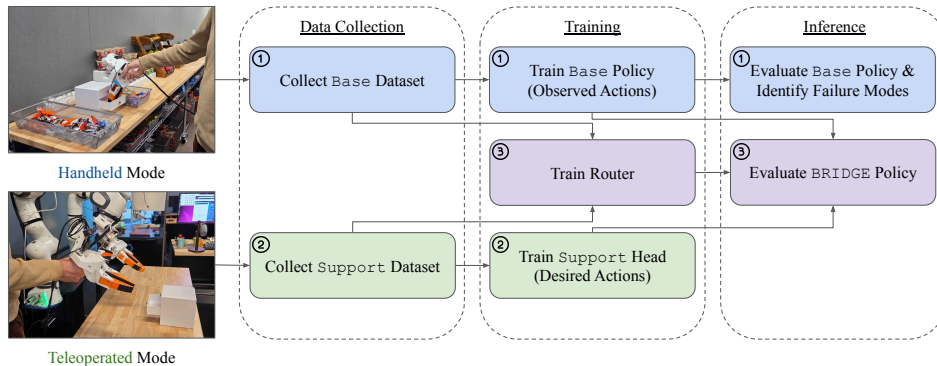


Figure 2: **Dual Mode Data Collection Pipeline.** First, we use DM-UMI in handheld mode to collect base demonstrations to learn the task scaffold. We then train and evaluate this base policy to identify failure modes. Second, we use DM-UMI in teleoperation mode to collect a targeted support dataset to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

small demonstration sets [14]. However, naively mixing heterogeneous human demonstrations can degrade performance compared to using a single source [15]. 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 [16] 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 [17, 18]. Similarly, intervention-based approaches keep a human-in-the-loop during data collection, reweighting demonstrations to emphasize corrections [19, 20, 21]. Unlike iterative approaches, Residual Policy Learning (RPL) [22, 23] composes a learned correction over a base controller. Recent work extends RPL to contact-rich manipulation under impedance control [24, 25]. However, these methods typically collect corrections with a model in the loop. In contrast, although our targeted demonstrations are informed by model failures, we collect our teleoperated examples offline. Our trained router selects between jointly trained handheld and teleoperated experts with an action-conditioned gate.

Mixture-of-Experts Policies. Mixture-of-Experts (MoE) architectures, which activate sparse sub-networks per input, have been instrumental in scaling language models [26]. Recently, sparse MoE diffusion policies [27, 28] 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. Furthermore, our router is guided by a non-parametric latent support signal, enforcing a hard-switch based on action validity.

3 BRIDGE: Gated Mixture of Experts for Contact-Rich Manipulation

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 jointly learns from both handheld and teleoperated supervision.

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 the DS80 camera pose and gripper encoders to teleoperate the robot end effector pose and gripper.

Dataset Collection. We use DM-UMI to collect both handheld and teleoperated demonstrations as shown in Figure 2. First, we use DM-UMI’s handheld mode to collect task demonstrations across diverse environments, thereby constructing a comprehensive base dataset. We train and evaluate a base policy on this data to identify empirical failure modes. Next, we use DM-UMI’s teleoperated mode to collect targeted demonstrations to capture embodiment-specific effects, including controller dynamics, robot kinematics, contact dynamics, and grasp stability. This approach leverages the complementary structure of both datasets: the base dataset (observed actions only) ensures broad 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 [29]. 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 [29, 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 [30] 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 [31] 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. [29] 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 average cosine similarity to the $k = 16$ neighbors in each bank,

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

We assign each latent with the label $\rho(z)$ to smooth the decision boundary,

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

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

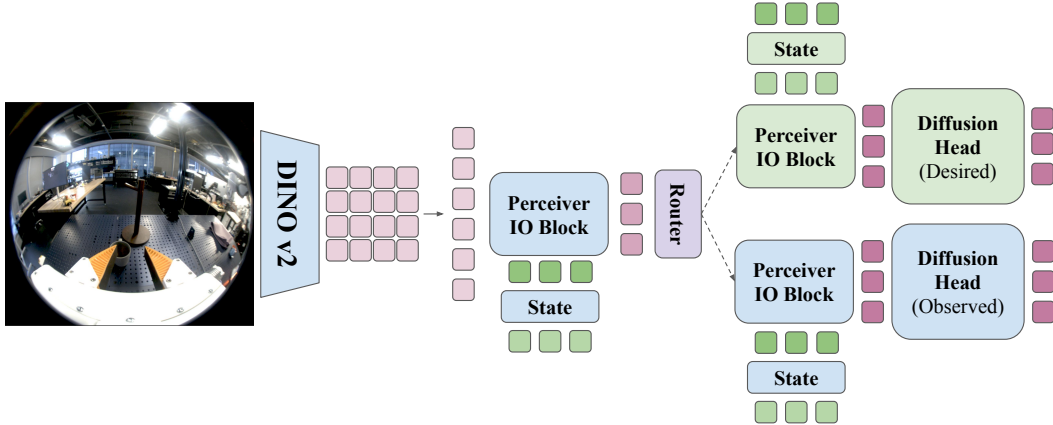


Figure 3: **Model Architecture.** We propose BRIDGE, an extension of Diffusion Policy that dynamically routes between predicting observed and desired actions. Visual observations are encoded via DINOv2 [30], processed through a Perceiver IO block, and fused with state features via cross-attention. This shared latent representation is passed to a state-conditioned router, which hard-switches between the observed diffusion head and the desired diffusion head. 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).

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,

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

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

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.

4 Experiments

In this section, 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.

Dataset. We summarize the number of demonstrations collected per task in Table 1. 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 suffice to significantly improve task success. The full-task teleoperation dataset was collected using approximately the same wall-clock budget as the corresponding base-plus-support dataset. We note that collecting the full-task

Table 1: **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%

[†]Full-task teleoperated demonstrations collected under an approximately time-matched data collection budget.

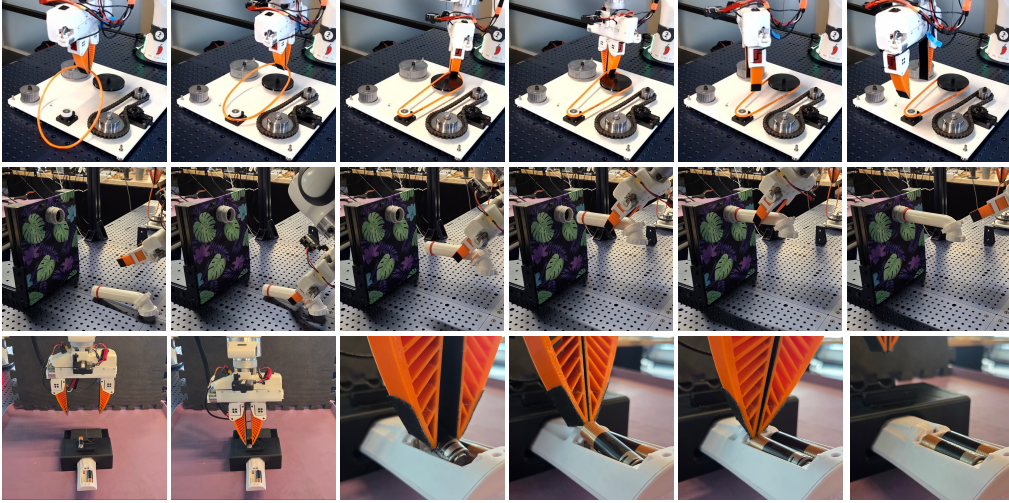


Figure 4: **Policy Rollouts.** We evaluate three precise, contact-rich tasks, including NIST pulley routing (*top*), pipe insertion (*middle*), and spring-loaded battery insertion (*bottom*).

teleoperated dataset was considerably more cognitively 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 between *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 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 3, differing only in their supervision strategy and training schedule:

- *Base Policy (Handheld).* We train the base policy (π_b) exclusively on handheld DM-UMI demonstrations (observed actions).
- *Base Policy (Teleoperated).* We train the base policy exclusively on teleoperated demonstrations (desired actions) as an upper bound.
- *Naive Mix.* We train the base policy on a combined dataset 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 3.

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 2, the Base Policy (Handheld) succeeds in 44.0% of trials, but its performance drops sharply in the top and bottom fixture positions. These configurations exposed a platform-level limitation; high-stiffness tracking frequently triggered Franka controller faults, while smoothing the commanded motion reduced faults at the cost of degraded tracking. Similar faults occurred for lower stiffness teleoperation and support data. We exclude faulted trials rather than counting them as policy failures. 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.

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%
BRIDGE (Ours)	4-4-4-4-3	19/25	76.0%
Base Policy (Teleoperated)	5-5-4-4-3	21/25	84%

Table 2: 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. Faulted trials caused by Franka controller errors were excluded and re-run to maintain 25 valid trials each.

Pipe Insertion Results. This task tests geometric precision as small pose errors easily cause jamming. As shown in Table 3, 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%, significantly closing the gap toward the teleoperated upper bound (63.3%).

Table 3: 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)	Score		
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%	

Spring-Loaded Battery Insertion Results. This task demonstrates that action mismatch is primarily localized to contact phases (Table 4). 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 a failure to insert the battery into the compartment. 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, proving that even unsuccessful trials are pushed much closer to completion (e.g., reaching the “Not Fully Inserted” stage). The remaining gap to full teleoperated success (40.0%) confirms that precisely regulating final contact forces remains the primary challenge.

Table 4: 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%

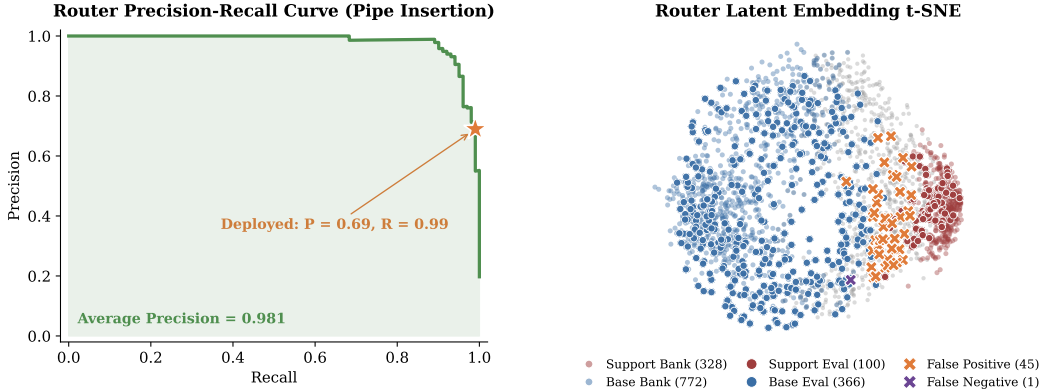


Figure 5: **Router Analysis.** We visualize the precision-recall curve for the pipe insertion task (left). The deployed router achieves 99.0% recall and 69.0% precision, favoring early support activation over missed handoffs. Computed t-SNE latent embeddings demonstrate clear separation between base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

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: the support bank covers the support-state distribution, naturally restricting successful routing to this manifold rather than requiring extrapolation to disjoint regions. We ablate hyperparameters used to train the router in the supplement.

Limitations and Future Work. Relying on handheld data collected in lab environments may underestimate the generalization potential of policies trained on large-scale, in-the-wild demonstrations. Although teleoperated policies perform the best in our evaluation, we posit that such policies lack robustness to out-of-distribution environments. Furthermore, our current approach for identifying task phases that require support data is empirical and future work should consider formalizing this such that it can be automated. Next, we also find that hard-switching introduces boundary sensitivity and control discontinuities during inference, suggesting a need for reference-governed or passivity-aware smoothing. Lastly, while allocating a separate refinement head per support region is highly interpretable, this architecture may scale poorly to long-horizon tasks with multiple contact regimes.

5 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 recovers up to 79% of the performance gap to the full teleoperation upper bound. 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 preprint 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 preprint 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 preprint arXiv:2304.13705*, 2023.
- [4] National Institute of Standards and Technology. Assembly performance metrics and test methods. <https://www.nist.gov/el/intelligent-systems-division-73500/robotic-grasping-and-manipulation-assembly/assembly>, 2026. Accessed: 2026-05-19.
- [5] O. X.-E. Collaboration. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint 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 preprint 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 preprint 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 preprint 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] R. Doshi, H. Walke, O. Mees, S. Dasari, and S. Levine. Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation. In *Conference on Robot Learning (CoRL)*, 2024.
- [11] L. Y. Chen, C. Xu, K. Dharmarajan, R. Cheng, K. Keutzer, M. Tomizuka, Q. Vuong, and K. Goldberg. Mirage: Cross-embodiment zero-shot policy transfer with cross-painting. In *Robotics: Science and Systems (RSS)*, 2024.
- [12] Z. Fu, Q. Zhao, Q. Wu, G. Wetzstein, and C. Finn. HumanPlus: Humanoid shadowing and imitation from humans. In *Conference on Robot Learning (CoRL)*, 2024.
- [13] 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 *Robotics: Science and Systems (RSS)*, 2024.
- [14] A. Mandlekar, 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 *Conference on Robot Learning (CoRL)*, 2023.
- [15] A. Mandlekar, 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 *Conference on Robot Learning (CoRL)*, 2021.

- [16] 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.
- [17] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.
- [18] 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 *Conference on Robot Learning (CoRL)*, 2021.
- [19] 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 *Robotics: Science and Systems (RSS)*, 2020.
- [20] A. Mandlekar, D. Xu, R. Martín-Martín, Y. Zhu, L. Fei-Fei, and S. Savarese. Human-in-the-loop imitation learning using remote teleoperation. *arXiv preprint arXiv:2012.06733*, 2020.
- [21] 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 *Robotics: Science and Systems (RSS)*, 2023.
- [22] T. Silver, K. Allen, J. Tenenbaum, and L. Kaelbling. Residual policy learning. *arXiv preprint arXiv:1812.06298*, 2018.
- [23] 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 *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.
- [24] 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.
- [25] Y. Huang, N. Ma, W. Zhao, Z. Liu, J. Sun, Q. Wang, and Y. Chen. Force-aware residual dagger via trajectory editing for precision insertion with impedance control, 2026. URL <https://arxiv.org/abs/2603.04038>.
- [26] 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, 2017. URL <https://arxiv.org/abs/1701.06538>.
- [27] C. Hao, X. Zhai, Y. Liu, and H. Soh. Abstracting robot manipulation skills via mixture-of-experts diffusion policies, 2026. URL <https://arxiv.org/abs/2601.21251>.
- [28] K. Guo, H. Liu, Y. Sun, R. Zhao, J. Zhou, and J. Ma. Moe-act: Scaling multi-task bimanual manipulation with sparse language-conditioned mixture-of-experts transformers, 2026. URL <https://arxiv.org/abs/2603.15265>.
- [29] 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. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.
- [30] 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 preprint arXiv:2304.07193*, 2023.
- [31] 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 preprint arXiv:2107.14795*, 2021.

A Implementation Details

We present the training, pre-processing, model, and optimization details for our baselines and method in Table 5.

Table 5: **Hyperparameters.** All training, pre-processing, model, and optimization details described below are used for all experiments unless otherwise specified.

Category	Setting	Value
Image Pre-Processing	Image Resize	518×518
	Training Augmentation	Jitter Brightness=0.1, Contrast=0.1, Saturation=0.05, Hue=0.05 with $p = 0.9$
	Evaluation Preprocessing	Resize-only
Observation & Action Setup	Action Horizon	24
	Action Step Size	3
	Image Horizon	1
	Image Step Size	3
	End-Effector Horizon	3
	End-Effector Step Size	3
	State components	SE3 Pose, Relative Orientation, Gripper Width
Dataset	Validation Split	10%
	Batch Size	64
Vision Backbone	Vision Encoder	DINOv2 ViT-S/14 (Pre-Trained)
	Latent Dimension	384
Diffusion Head	Training Timesteps	50
	Inference Timesteps	16
	U-Net Condition Type	FiLM
	Conditioning Dimension	384×4
	Base U-Net Down Dims	[128, 256, 512]
	Support U-Net Down Dims	[96, 192, 384]
	Kernel Size	5
	Number of Groups	8
	Diffusion-step Embedding Dim	32
	β Schedule	Squared Cosine Capped v2
	β_{start}	10^{-4}
β_{end}	0.02	
Optimization	Optimizer	AdamW
	Base LR	5×10^{-4}
	Vision Encoder LR	5×10^{-5}
	Adam β	(0.95, 0.999)
	Adam ϵ	10^{-8}
	Weight Decay	10^{-6}
	Warmup Iterations	1000
Training	Max Epochs	300
	Precision	bf16 Mixed
	Gradient Clipping	1.0

Router Architecture. The router uses a lightweight residual MLP adapter on top of the policy conditioning features. We concatenate the visual features with the available state features and project them to a chosen latent dimension with a linear layer. We then apply layer normalization followed by a two-layer MLP with GELU activation. The hidden dimension is set to the conditioning dimension used for the diffusion head, $d_{\text{cond}} = 384$. The final linear layer of the adapter is zero-initialized, so the adapter begins as an identity-style residual update and learns deviations from the pretrained conditioning representation during router training.

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 1. Averaging actions from these experts can therefore produce intermediate commands that are not

meaningful for either mode. Although hard switching can introduce boundary sensitivity, we reduce abrupt transitions using the downstream control stack and temporal filtering. 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 rollout, we further debounce the router by activating the support expert only when $G_\psi(z_t) > \eta$, with $\eta = 0.5$, for a continuous duration of 0.25 s. This prevents transient spikes in the router output from spuriously activating the support expert.

Image Masking. To prevent the model from exploiting the visual differences in the body of the on-robot gripper (Figure 6 a) and its handheld data collection counterpart (Figure 6 b), we mask the gripper body (Figure 6 c).

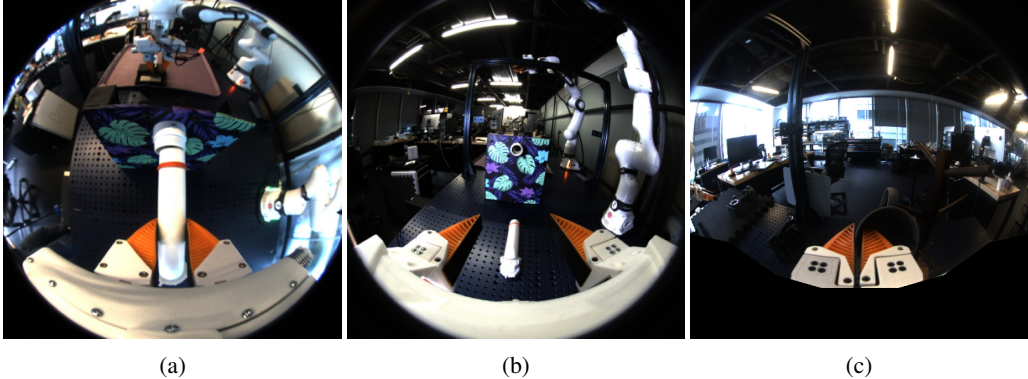


Figure 6: **Image Masking.** We visualize the on-robot gripper (a) alongside its handheld data collection counterpart (b). To prevent the model from exploiting the visual differences between the bodies of these two devices, we apply a mask to the gripper body (c).

B Dual-Mode System Characterization

We evaluate the DS80’s pose tracking accuracy in Table 6. 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.

Table 6: **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 translation error, relative translation error, absolute rotation error, and relative rotation 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

On average, we find that DM-UMI demonstrations are faster than teleoperated ones (Table 7). Across all evaluated tasks, handheld data collection was about $3\times$ faster, and collecting partial teleoperated demonstrations was about $2\times$ faster than collecting full-task teleoperation, respectively. These runtime statistics still underestimate the practical cost of full teleoperation which is more

Table 7: **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.

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

physically tiring, and is more prone to robot faults during long contact-rich segments. Thus, even when full teleoperation provides a strong in-domain upper bound, targeted support provides a more practical way to collect desired action supervision.

C Preliminary Exploration of Data Scaling

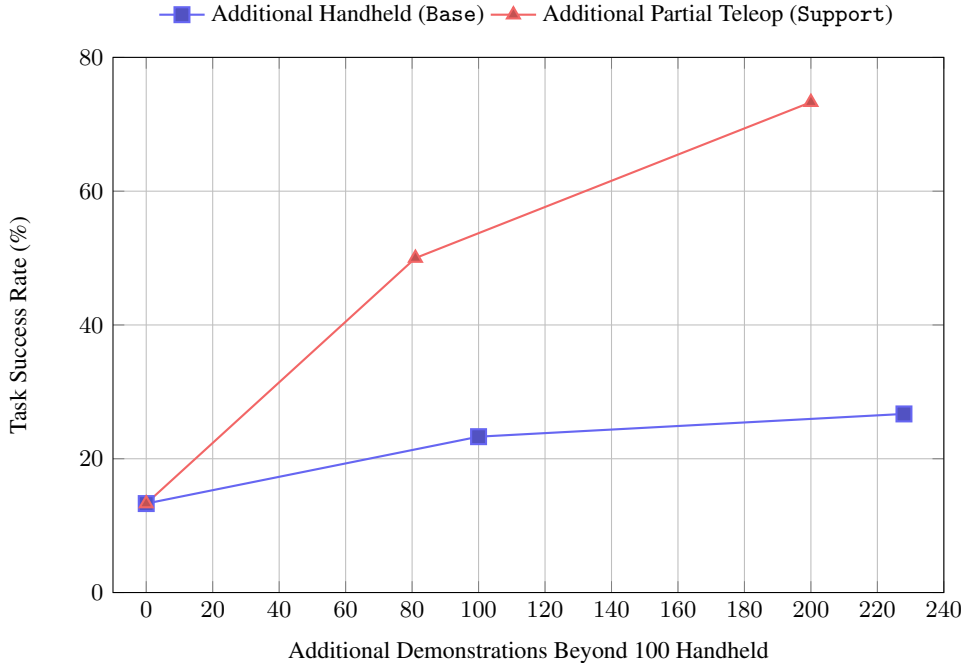


Figure 7: **Impact of Scaling Targeted Support Data.** Starting from the same 100-demo base policy from the main paper, incorporating targeted partial teleoperation demonstrations improves pipe insertion success substantially more than simply adding more handheld demonstrations. While adding over 120 additional handheld demonstrations yields only marginal improvements, utilizing our targeted partial teleoperation approach drastically increases task success to over 70%.

We investigate the impact of handheld and targeted teleoperated demonstrations on the pipe insertion task. Table 8 summarizes these initial scaling results.

Teleoperated Upper Bound. 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 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.

Scaling Handheld Data. Increasing the number of handheld demonstrations alone provides only 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

Table 8: **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%

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 nominal UMI behavior, but rather imperfect trajectory tracking under contact-rich regimes.

Scaling Targeted Support Data. 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 performance improvement primarily stems from converting “aligned-but-not-inserted” states into full insertions, confirming that targeted support data effectively provides the necessary corrective actions to overcome small misalignments during contact.

Limitations of BRIDGE. These results also highlight a structural limitation of BRIDGE. Because the support expert is invoked only after the base policy brings the robot close to the contact region, overall performance is bottlenecked by the quality of the base trajectory. If the initial grasp is poor or the base policy approaches the fixture from an awkward pose, the support expert is forced to recover from an inherently more difficult state. Consequently, while support data is highly leveraged, it is not strictly independent of the upstream base policy. Overall, these scaling results do not suggest that BRIDGE outperforms full teleoperation when such data is abundant. Rather, they demonstrate that when leveraging scalable handheld demonstrations, providing targeted desired action support is significantly more effective than simply increasing observed action handheld data. While additional handheld demonstrations provide a broader distribution of nominal trajectories, they still encode observed behaviors that the robot struggles to track precisely under contact. In contrast, support demonstrations directly supervise the specific task regions where the base policy fails.

Practical Cost of Full-task Teleoperation. The cost of full-task teleoperation is highly task-dependent and is not fully captured by wall-clock collection time. For short, spatially compact tasks like pipe insertion, experienced operators can 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, full-task teleoperation requires the operator to maintain O-ring tension over a wide range of motion without direct force feedback. We found this substantially more difficult: the robot frequently triggered controller faults, and the operator was forced into awkward postures for extended periods. 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 is competitive in wall-clock time for shorter tasks, targeted support collection minimizes the physical burden of continuous teleoperation and eliminates the redundancy of repeatedly teleoperating trivial free-space phases.

Cross-Robot Teleoperation Transfer. The 120 demonstration full teleoperation policy represents an in-domain upper bound rather than evidence that full teleoperation is a scalable replacement for

targeted support data. We validate the effectiveness of our approach through cross-robot transfer by evaluating our pipe insertion policy trained on Franka Emika A on a second robot, Franka Emika B. Under this cross-robot transfer, performance dropped sharply from 93.3% to 3.3% success, with most trials reaching only partial alignment. This outcome aligns with the known sensitivity of contact-rich policies to variations in robot calibration, controller dynamics, and minor setup biases. We also conducted a preliminary adaptation test by augmenting the handheld base policy with 15 minutes of targeted support data (50 demonstrations) collected on Franka Emika B. In this setting, BRIDGE achieved a total success score of 7.00/10, consisting of 3 full insertions and 4 partial insertions (including 2 partial insertions and 2 aligned-but-not-inserted outcomes). While further investigation is required, this preliminary result suggests that targeted support data collection may be a cost-effective way to reuse handheld data and recover cross-robot performance without recollecting full-task teleoperation demonstrations.

D Impact of Using Spatial Tokens vs. CLS Token for Visual Conditioning

We utilize the fully teleoperated dataset and the pipe insertion task to evaluate whether a visual bottleneck in the policy affects contact-rich insertion performance. While the CLS token provides a compact global summary of the DINOv2 features, pipe insertion depends on the fine spatial relationship between the pipe and the fitting. Small local changes in position or orientation may require distinct corrective actions, even when the overall scene appears unchanged. We hypothesize that relying solely on the CLS token causes states requiring different insertion corrections to map to similar latent representations, thereby limiting performance.

To mitigate this, we incorporate *all* spatial tokens to preserve local geometric information from the visual backbone. Because DINOv2 produces a large number of patch tokens, we employ a Perceiver-IO-style query adapter to compress the spatial token sequence before passing it to the policy. As shown in Table 9, this representation significantly improves performance across two different data scales. With 60 fully teleoperated demonstrations, spatial tokens improve the task success rate from 43.3% to 63.3% and raise the overall score from 21.50/30 to 23.75/30. With 120 demonstrations, the CLS-only policy achieves 60.0% success, whereas the spatial token policy reaches 93.3%. This suggests that the full-teleoperation baseline is constrained not only by demonstration count but also by the visual representation used to condition the policy.

Furthermore, the spatial-token adapter markedly improves data efficiency. Trained on just 60 demonstrations, the spatial-token policy (63.3% success, 23.75/30 score) outperforms the CLS-only policy trained on twice as much data (60.0% success, 24.00/30 score). This demonstrates that preserving local visual geometry reduces the number of teleoperated demonstrations required to learn precise insertion behaviors. This benefit scales with additional data; at 120 demonstrations, spatial-token conditioning reaches a near-perfect score of 29.00/30.

Table 9: **Effect of Spatial Tokens on Performance.** 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 (60 demos, CLS Only)	7-3-3	0-3-1	3-3-5	0-1-1	21.50/30	43.3%
Base Policy (60 demos, Spatial Tokens)	3-7-9	1-1-1	5-0-0	1-2-0	23.75/30	63.3%
Base Policy (120 demos, CLS Only)	5-5-8	0-0-0	5-5-2	0-0-0	24.00/30	60%
Base Policy (120 demos, Spatial Tokens)	9-10-9	0-0-0	1-0-1	0-0-0	29.00/30	93.3%

E Desired vs. Observed End-Effector Tracking Under Contact

Figure 8 provides the real-system counterpart to the illustrative trade-off in Figure 1 of the main paper. Our central claim concerns *action validity*: an observed (handheld) trajectory is not a valid

desired trajectory in contact-rich phases. Because this is a kinematic claim, we demonstrate it directly in the position domain using the controller’s own logs from a NIST pulley routing run, comparing the commanded (desired) end-effector pose x_d against the achieved (observed) pose x .

In free-space phases, the controller closely tracks the commanded pose, and the observed pose is a faithful proxy for the desired one. Underloaded contact, however, the commanded pose is not achieved, and a persistent gap $\Delta = x_d - x$ opens between the desired and observed trajectories, closing again only once the load is released. This gap is the signature of a regime in which the observed trajectory was never a viable desired trajectory, so no observed-only label can recover the commanded intent. Under an impedance controller, this same gap is what is converted into contact force via $F = K\Delta$; the peak in-contact gap of roughly 53 mm at a teleoperation-level stiffness of $K_{\text{trans}} = 1000 \text{ N/m}$ corresponds to a derived contact force of order $K\Delta \approx 53 \text{ N}$. We therefore treat contact force as a derived consequence of the measured desired-observed mismatch rather than a separately instrumented quantity.

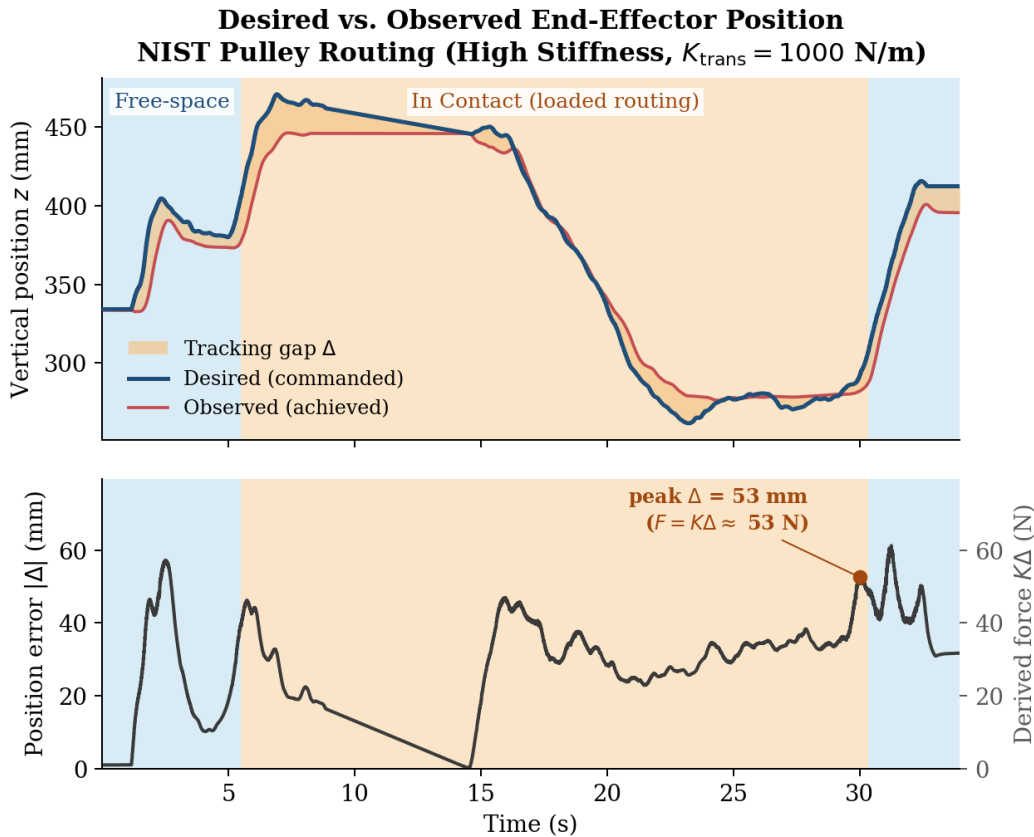


Figure 8: **Desired vs. Observed End-Effector Position during NIST pulley routing.** *Top:* the commanded (desired, x_d) and achieved (observed, x) vertical end-effector position for a single routing run on the real system. In free-space phases (blue) the controller tracks the commanded pose closely; under loaded contact (orange) a persistent tracking gap $\Delta = x_d - x$ (amber) opens and closes only after the load is released, indicating that the observed trajectory is not a valid desired-trajectory label in contact. *Bottom:* the position-error magnitude $|\Delta|$, with the corresponding derived contact force $K\Delta$ (impedance law $F = K\Delta$) on the right axis. The peak in-contact gap ($\Delta \approx 53 \text{ mm}$ at $K_{\text{trans}} = 1000 \text{ N/m}$) corresponds to a derived contact force of $K\Delta \approx 53 \text{ N}$.

F Qualitative Visualization of Task Failure Modes

We present a qualitative analysis of the most frequently observed failure modes across all three manipulation tasks: the NIST Pulley, Battery Insertion, and Pipe Insertion (Fig. 9). Common failure modes primarily stem from lateral and planar misalignments, unstable or weak grasps, and unforeseen collisions with the task environment.

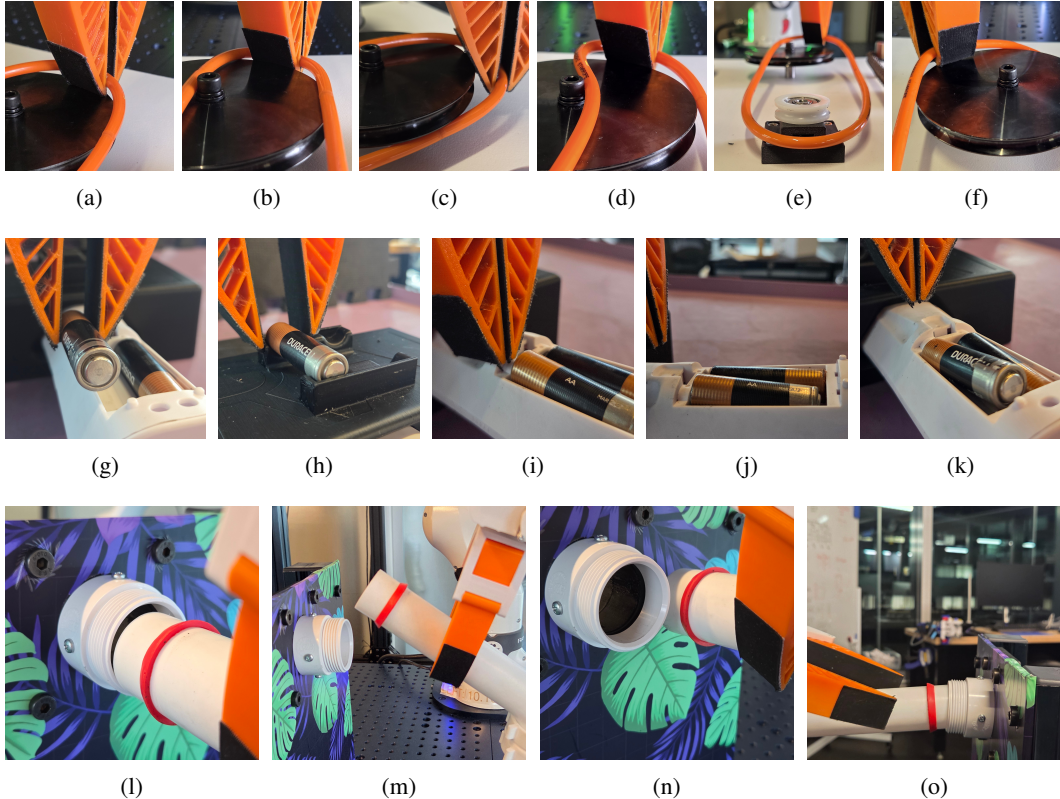


Figure 9: **Visualization of Common Failure Modes Across all Evaluated Tasks.** (a–f) *NIST Pulley Task*: Failures typically arise from (a) incomplete pulley clearance due to gasket tension, (b) imprecise positioning during lowering, (c) overshooting the groove entirely, (d) binding on the bolt after lateral misalignment, (e) improper seating on the smaller pulley, and (f) partial or incomplete insertion into the groove. (g–k) *Battery Insertion*: Insertion errors commonly include (g) general lateral misalignment, (h) grasp failure and dropping due to collision with the holder, (i) applying downward force to the remote chassis instead of the battery slot, (j) incomplete seating within the compartment, and (k) missing the slot entirely at the far end of the battery. (l–o) *Pipe Insertion*: Typical failure modes consist of (l) significant planar misalignment leading to a drop upon gripper release, (m) in-hand pivoting caused by a weak grasp, (n) missing the target opening entirely due to poor alignment, and (o) jammed incomplete insertions where the insertion fails but does not fall out.

G Ablation on Router Hyperparameters

We ablate router performance using a manually labeled held-out evaluation set collected specifically for router analysis. Therefore, the absolute performance numbers differ from the results reported in the main paper. We use this set strictly for relative comparisons between pseudo-labeling variants. Because support demonstrations are collected as local corrective segments rather than full end-to-end task rollouts, there is no one-to-one temporal correspondence between base and support trajectories. Therefore, these annotations should be interpreted as approximate support-region labels rather than exact ground-truth switch labels. We label these regions based on the observed task phase and whether the state falls within the intended corrective regime.

Filtering for KNN Pseudo-Labeling, ϵ . During BRIDGERouter training in Stage 3, we avoid ambiguous pseudo-labels in regions where base and support demonstrations overlap by removing base samples within an ϵ -neighborhood of support latents before constructing the training set. This filtering is crucial because support and non-support banks often overlap in the latent space; visually similar observations can occur in different task phases, requiring entirely different actions. Without this step, the non-support bank retains support-like negatives, leading to inconsistent KNN pseudo-labels. In our implementation, the base policy state vector includes the robot’s pose relative to the start of the episode. Because support demonstrations are collected from intermediate states, their episode start positions differ from those of full-task base demonstrations. During inference, only the full-task start pose is initially available; the support policy start pose becomes available only after a support region is identified, and the router switches to the support expert. When encoded by the trained base policy latent generator, this relative pose convention makes support samples highly distinguishable from nominal base samples. We treat this relative pose structure as privileged information, using it strictly to improve offline KNN pseudo-label performance. The deployed router does not use the support episode start pose in its observation space.

Table 10 ablates the cosine-similarity threshold (ϵ) used to filter non-support bank candidates. This threshold dictates the quality and coverage of the negative bank. Very strict thresholds (≤ 0.0) remove all non-support candidates, making KNN pseudo-labeling ill-defined, while overly permissive thresholds ($\epsilon = 1.0$) retain support-like negatives, severely degrading support recall. The intermediate threshold of $\epsilon = 0.75$ yields the best trade-off in both privileged and non-privileged settings. Incorporating privileged relative pose information shifts the failure mode at permissive thresholds: at $\epsilon = 1.0$, support recall improves from 2.3% to 31.4%. This suggests that task-phase information partially disambiguates support regions, though privileged pose alone does not eliminate the need for overlap filtering.

Table 10: **KNN Bank Filtering Ablation.** We ablate the impact of the similarity filtering threshold (ϵ) on classification performance across both vision latent and pose augmented bank features. The ϵ threshold controls the retention of non-support candidates based on their similarity to support latents. Stricter thresholds (≤ 0.0) completely remove the non-support bank, whereas highly permissive thresholds (1.00) retain too many support-like negatives, heavily degrading recall. An intermediate threshold of $\epsilon = 0.75$ provides the optimal trade-off for both feature sets, maximizing overall accuracy (88.2% for vision latent and 81.8% for pose augmented) by effectively balancing support precision and recall.

Bank Features	ϵ	Non-Support Bank Size	Accuracy	Support Precision	Support Recall
Vision Latent	-1.0, -0.5, 0.0	0	-	-	-
	0.50	1	46.0%	46.0%	100.0%
	0.75	896	88.2%	95.7%	77.9%
	1.00	1000	55.1%	100.0%	2.3%
Pose Augmented	-1.0, -0.5, 0.0	0	-	-	-
	0.50	24	65.2%	57.0%	100.0%
	0.75	986	81.8%	96.4%	62.8%
	1.00	1000	67.4%	93.1%	31.4%

Router Inference Threshold, η . Table 11 demonstrates that the router can learn useful support regions even from unprivileged KNN labels, achieving 91.7% support precision and 64.0% support recall. However, upgrading to privileged, pose-augmented KNN labels significantly boosts overall accuracy to 93.6% and increases support recall to 96.5% while maintaining high precision. Ablating the inference threshold (η) on held-out pipe insertion data confirms this performance leap is not merely the result of a favorable operating point. The router trained on privileged labels proves highly robust, maintaining support recall above 95% across all thresholds between 0.1 and 0.9, while precision scales from 89.4% to 92.1%. In contrast, the unprivileged router yields consistently lower recall, dropping from 67.4% at $\eta = 0.1$ to 59.3% at $\eta = 0.9$.

This disparity highlights that while privileged relative pose is not strictly required for deployment, it provides a fundamentally cleaner offline teacher signal that allows the vision-only router to map a much more stable support boundary. When analyzing these results, it is important to note that the non-parametric KNN bank (the teacher) and the trained parametric router (the student) are not expected to perform identically. The KNN bank generates pseudo-labels based on local nearest-neighbor structures, bank filtering, and distance metrics. The trained router then distills these pseudo-labels into a smoother decision boundary using only inference-time visual observations. Ultimately, the KNN results reflect the raw quality of the teacher’s signal, whereas the router’s results demonstrate how effectively a deployable, vision-only router can learn from that supervision.

Table 11: **Prediction-Threshold Sweep for Vision-Only Router.** We perform a parameter sweep of the inference-time router decision threshold (η) for a trained vision-only router, comparing the effect of training with pseudo-labels derived from a vision latent KNN vs. a pose augmented KNN. Utilizing privileged pose-augmented KNN pseudo-labels yields a significantly more robust router. Across all evaluated thresholds, the pose-augmented source achieves consistently higher overall accuracy (exceeding 93%) and substantially improves support recall (over 95%) compared to the vision-latent baseline, resulting in a marked reduction of false negatives. We report true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN), where a positive denotes a state labeled as belonging to a support region.

Pseudo-label Source	Prediction Threshold (η)	Accuracy	Support Precision	Support Recall	Confusion TP/FP/TN/FN
Vision Latent KNN	0.1	82.4%	92.1%	67.4%	58 / 5 / 96 / 28
	0.3	81.3%	91.8%	65.1%	56 / 5 / 96 / 30
	0.5	80.7%	91.7%	64.0%	55 / 5 / 96 / 31
	0.7	80.7%	91.7%	64.0%	55 / 5 / 96 / 31
	0.9	78.6%	91.1%	59.3%	51 / 5 / 96 / 35
Pose Augmented KNN	0.1	93.6%	89.4%	97.7%	84 / 10 / 91 / 2
	0.3	93.0%	89.2%	96.5%	83 / 10 / 91 / 3
	0.5	93.6%	90.2%	96.5%	83 / 9 / 92 / 3
	0.7	94.1%	91.2%	96.5%	83 / 8 / 93 / 3
	0.9	94.1%	92.1%	95.3%	82 / 7 / 94 / 4

H Visualization of Support Regions

Figure 10 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 15% of the total trajectory length in the battery insertion task, 4% in the pipe insertion task, and 8% in the NIST pulley routing task.

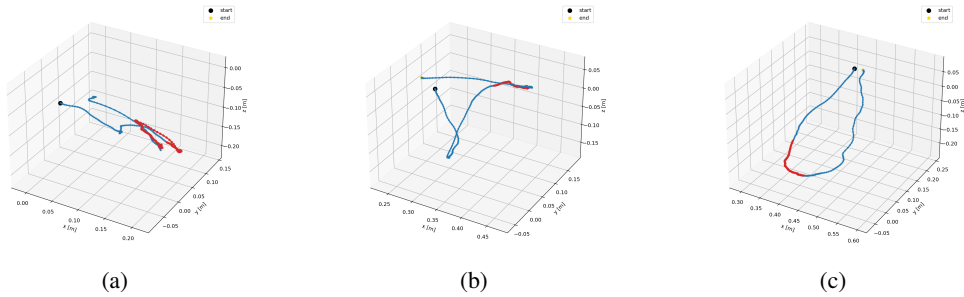


Figure 10: **Trajectory Support Coverage Across Tasks.** As illustrated by the red highlighted segments, the median trajectory support coverage spans 15% of the trajectory length for battery insertion (a), 4% for pipe insertion (b), and 8% for NIST pulley routing (c).