DEMYSTIFYING ACTION SPACE DESIGN FOR ROBOTIC MANIPULATION POLICIES

Yuchun Feng[1*], **Jinliang Zheng**[1,2*], **Zhihao Wang**[1,3], **Dongxiu Liu**[1], **Jianxiong Li**[1], **Jiangmiao Pang**[2], **Tai Wang**[2†], **Xianyuan Zhan**[1†]
[1] AIR, Tsinghua University [2] Shanghai AI Lab [3] Peking University
[*] Equal Contribution [†]Corresponding Author
wangtai@pjlab.org.cn & zhanxianyuan@air.tsinghua.edu.cn

ABSTRACT

The specification of the action space plays a pivotal role in imitation-based robotic manipulation policy learning, fundamentally shaping the optimization landscape of policy learning. While recent advances have focused heavily on scaling training data and model capacity, the choice of action space remains guided by ad-hoc heuristics or legacy designs, leading to an ambiguous understanding of robotic policy design philosophies. To address this ambiguity, we conducted a large-scale and systematic empirical study, confirming that the action space does have significant and complex impacts on robotic policy learning. We dissect the action design space along temporal and spatial axes, facilitating a structured analysis of how these choices govern both policy learnability and control stability. Based on **13,000+** real-world rollouts on a bimanual robot and evaluation on **500+** trained models over four scenarios, we examine the trade-offs between absolute vs. delta representations, and joint-space vs. task-space parameterizations. Our large-scale results suggest that properly designing the policy to predict delta actions consistently improves performance, while joint-space and task-space representations offer complementary strengths, favoring control stability and generalization, respectively.

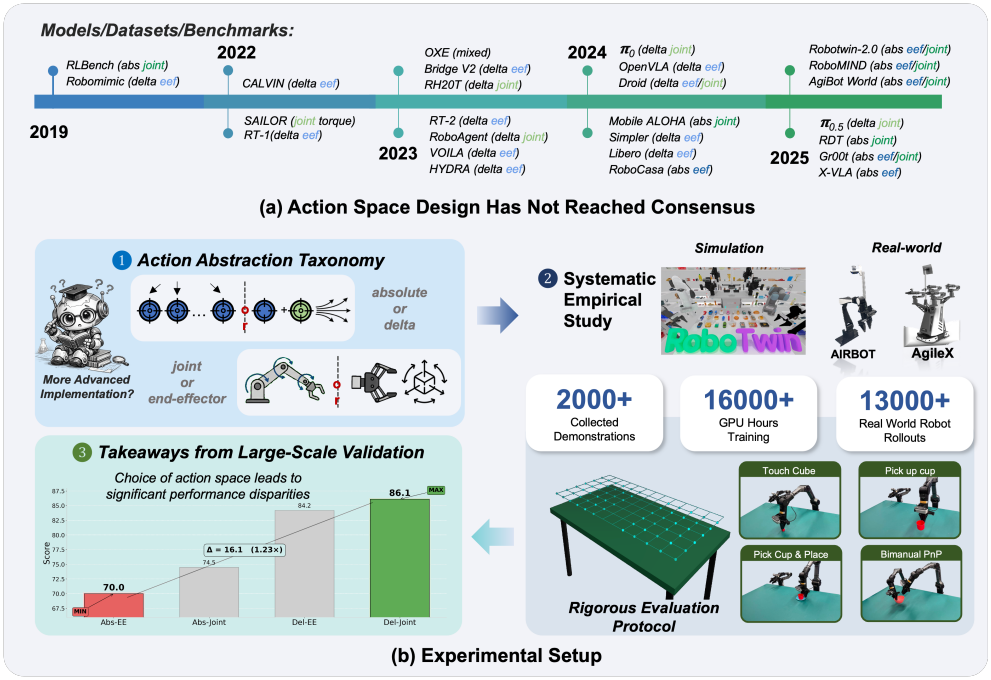


Figure 1: Overview of our study on action space design. (a) Historical analysis shows the divergent usage of action spaces (Absolute vs. Delta, Joint vs. EEF) in existing literature. (b) Our experimental setup includes an action abstraction taxonomy and a large-scale benchmark on both simulation and real-world platforms. We invest over 13,000 real-world rollouts to quantify the impact of these design choices, revealing significant performance gaps and identifying best practices for robotic manipulation under various scenarios.

1 INTRODUCTION

Learning-based robotic manipulation policies have achieved remarkable progress in recent years, evolving from simple pick-and-place to dexterous, precision-critical tasks Brohan et al. (2022; 2023a); Chi et al. (2023); Zhao et al. (2023); Jang et al. (2022). While recent advances largely focus on scaling training data and model capacity NVIDIA et al. (2025); Black et al. (2025); Lin et al. (2025a), the specification of *action space*, which is the underlying interface bridging neural predictions and physical hardware, remains an overlooked yet critical determinant of success. As the primary supervision signal, the choice of action representation governs not only the learnability of the policy but also the stability of deployment Eßer et al. (2024); Zheng et al. (2025). Subtle changes in this interface can drastically alter the optimization landscape, or even distinguish a robust policy from one that fails to generalize Chi et al. (2023). Although important, as illustrated in Figure 1(a), the research community still has no consensus on the best practices of action space design over the past years. Historically, end-effector pose was favored for its semantic simplicity Liu et al. (2024), yet recent trends have pivoted toward joint-space representations to bypass the numerical instabilities of *inverse kinematics* Black et al. (2025); Chen et al. (2025). Crucially, the design space extends far beyond simple spatial parameterization. It also involves a combinatorial explosion of choices, including temporal representation (absolute vs. relative) and prediction horizons (e.g., action chunking). Currently, the field lacks a consensus or a unified understanding to navigate through these numerous choices. Researchers often rely on ad-hoc heuristics or legacy configurations inherited from different codebases, leading to a fragmented landscape where "state-of-the-art" results are often conflated with specific, undocumented control choices Bjorck et al. (2025). Such ambiguity not only impedes reproducibility but also hampers the development of foundation models.

Addressing this ambiguity is crucial for guiding the design of future robotic manipulation policies. While prior works Chi et al. (2023); Eßer et al. (2024) have provided some initial insights, deriving comprehensive and reliable guidance for action space selection remains a non-trivial challenge. The difficulty stems from the substantial heterogeneity of robotic learning settings, the limited fidelity of simulation environments, and the high cost of real-world robotic evaluation. Consequently, existing studies are often limited in their empirical scope and lack systematic comparisons. As the field advances toward large-scale generalist robot models Team et al. (2025), the cost of suboptimal control interface design becomes increasingly consequential, underscoring the need for principled and unified design guidelines. To bridge this gap, we present the first large-scale, systematic empirical study that investigates how action space design impacts robotic policy learning. We begin by formalizing and dissecting action space design along two orthogonal axes: a temporal axis (absolute vs. delta parameterization, action chunking) and a spatial axis (joint-space vs. task-space control). We investigate how these design choices induce fundamental trade-offs between policy learnability, control stability, and deployment performance.

Building on this foundation, we conduct a comprehensive, large-scale experimental study across three platforms: the recently advanced simulation RoboTwin-2.0 Chen et al. (2025) and two real-world robotics platforms, AgileX PiPER and AIRBOT, all under stringent evaluation protocols. Specifically, the main experiments focus on AgileX, where we design a benchmark suite consisting of four diverse tasks, ranging from precision-critical sanity checks to complex dynamic bimanual coordination. To ensure statistical significance and fair comparison, we introduce a grid-based spatial coverage strategy that standardizes initial conditions across all trials. Through extensive real-world experiments, we first identify the most effective implementation details for each action space design via targeted preliminary studies. We then perform controlled grid searches along several critical axes, including data scale, model expressiveness, and training duration. Overall, this study constitutes a substantial empirical undertaking, comprising over **2,000** collected demonstrations and more than **13,000** real-world rollouts across **500+** trained models. The main results, together with five carefully designed cross-validation experiments, reveal two core insights:

Summary

Temporal abstraction is decisive: *delta-based* action representations consistently outperform *absolute* actions across all learning paradigms when implemented with appropriate practices.
Spatial abstraction is setting-dependent: *Joint-space* control excels in standard scenarios with sufficient data and extensive training, while *task-space* representations demonstrate superior performance in generalized settings like cross-embodiment and transfer learning.

2 ACTION ABSTRACTION TAXONOMY

We consider the problem of imitation learning for robotic manipulation, formulated as learning a policy π_θ from expert trajectories $\mathcal{D} = \{\tau_j\}_{j=1}^{M},\ \tau_j = \{(o_t, a_t)\}_{t=1}^{N_j}$ that produces executable actions a_t conditioned on observations o_t at timestep t. While this formulation is standard Li et al. (2025b), the physical realization of an action can vary substantially depending on the control interface exposed to the policy. To reason about these differences in a principled manner, we decompose the action representation space along two orthogonal axes: *spatial abstraction* and *temporal abstraction*, as illustrated in Figure 2. We begin by formalizing these abstractions and their impact on learning. Subsequently, we examine *action chunking*, a pivotal technique in modern architectures, and its interplay with action space design (See Appendix G for detailed formalizations).

2.1 SPATIAL ABSTRACTION

Spatial Abstraction defines the abstraction boundary between the learned policy and the hardware controller. At the lowest level of spatial abstraction lies the actuator space (e.g., motor torques or currents), which directly governs the robot's physical dynamics. However, direct torque-level supervision is rarely adopted for high-level manipulation policies, as it suffers from high dimensionality and poor sample efficiency Yu et al. (2025). Accordingly, we restrict our scope to the two dominant kinematic abstractions used in practice: *configuration space* (joint positions) and *task space* (robot-based end-effector pose). While joint space and task space are kinematically equivalent via forward and inverse kinematics solvers, the choice of supervision domain induces distinct optimization landscapes. Task-space control provides a geometrically meaningful abstraction that aligns naturally with object-centric visual observations. However, it relies on inverse kinematics solvers during deployment, which introduces numerical singularities and error accumulation that can degrade execution robustness Lee (1982). In contrast, joint-space control avoids solving inverse kinematics, but this robustness comes at the cost of increased learning complexity: the policy must implicitly learn the robot's kinematic structure, mapping visual inputs onto a highly non-linear configuration manifold. Consequently, spatial abstraction presents a fundamental trade-off between learning alignment and execution robustness.

2.2 TEMPORAL ABSTRACTION

The second axis of analysis, orthogonal to the spatial axis, is *temporal abstraction*, which specifies the order of temporal derivatives represented by the predicted action sequence. At one end of the spectrum is *absolute* representation (0th-order), which specifies target states directly. At the other end are *relative* or *delta* representations (1st-order), which specify state increments. It is worth noting that we adopt a position-based low-level controller as the interface, aligning with standard practices in the community Black et al. (2025). Consequently, our "1st-order" formulation refers to the *semantic meaning* of the policy output rather than the physical control mode, decoupling high-level motion planning from low-level dynamic regulation. Although higher-order formulations like force control are possible, they generally rely on accurate inertial modeling and significantly increase system complexity Yu et al. (2025). Consequently, we focus our analysis on absolute (0th-order) and delta (1st-order) representations, which governs a fundamental trade-off between learning stability and control accuracy. Under the absolute parameterization, the policy must map observations to global target states. This interface can encourage intuitive and precise grounding; however, it also requires the model to internalize complex real-world geometry and to cope with highly variable target distributions Chi et al. (2023); Liu et al. (2025b), thereby inducing substantial learning difficulty. In contrast, a delta parameterization predicts relative increments, yielding a better-conditioned and closed-loop learning target. However, deploying delta actions makes the system more sensitive to feedback imperfections: noise, latency, and tracking errors can accumulate over time and lead to drift Zhang et al. (2025a).

2.3 ACTION CHUNKING

Throughout this work, we adopt *Action Chunking* Zhao et al. (2023) as a default component. This technique has emerged as a cornerstone for action space shaping. By predicting a sequence of future actions, policies can better capture temporal dependencies, leading to substantial performance gains Zhang et al. (2025a). However, we identify that the introduction of chunking creates a non-

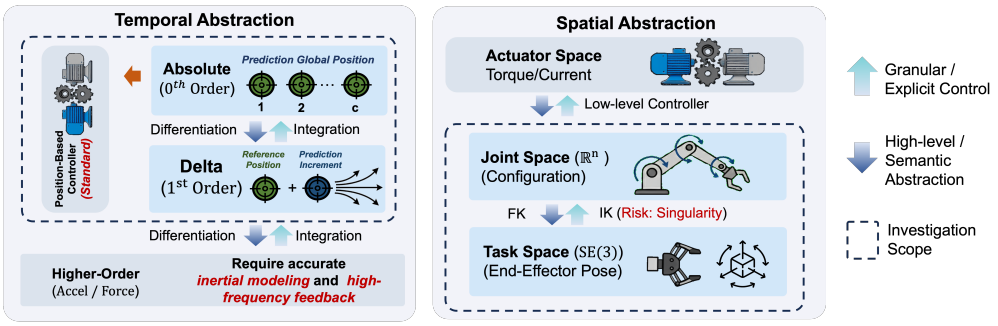


Figure 2: Hierarchy of the action space for robotic manipulation policies and its abstraction taxonomy

trivial structural ambiguity, resulting in two critical design challenges that remain under-explored in the robotics literature:

1. Ambiguity in Delta Alignment. Integrating chunking with delta actions necessitates a choice of reference frame: *step-wise delta* (relative to the immediately preceding predicted state within the sequence) Liu et al. (2024); Mees et al. (2022) versus *chunk-wise delta* (relative to the robot's state at the start of the chunk) Black et al. (2025). This choice fundamentally reshapes the action distribution.

2. Horizon-Abstraction Coupling. While the chunking horizon k is typically optimized as an isolated hyperparameter, we hypothesize a fundamental coupling between k and the choice of action abstraction. In specific, delta-based control may necessitate shorter horizons to facilitate rapid correction, whereas absolute position control might benefit from longer horizons to maintain global spatial grounding. Elucidating the interplay between these factors is a prerequisite for designing a proper action space for policy learning.

3 Experimental Setup

The goal of this study is to derive **practical and generalizable guidelines** for action space design. To ensure that these guidelines are robust across different scenarios, we conduct a large-scale empirical investigation spanning multiple hardware platforms, task configurations, and learning regimes. In this section, we summarize the common experimental components used throughout the paper, including model variations, hardware setups and evaluation protocol.

3.1 Model Architectures for Policy Learning

We aim to establish guidelines across a spectrum of paradigms, from specialized architectures like ACT Zhao et al. (2023) and Diffusion Policy Chi et al. (2023), to foundation models like π_0 Physical Intelligence et al. (2025).

Following common practice Brohan et al. (2022), we design a base architecture for policy learning. It uses a FiLM-conditioned ResNet-18 vision encoder Perez et al. (2018) paired with a 6-layer Transformer decoder. Further details of the implementation are provided in Appendix D. To investigate the interplay between action space design and model expressiveness, we adopt two prominent generative modeling paradigms, resulting in the following two model variants, which correspond to implementation in ACT Zhao et al. (2023) and DP Chi et al. (2023), respectively: **1. Regression-based Policy** is optimized using a standard Mean Squared Error (MSE) loss:

$$\mathcal{L}_{\mathrm{R}} = \mathbb{E}_{(\mathbf{o},\mathbf{a})\sim\mathcal{D}}\left[\left|\pi_\theta(\mathbf{o}) - \mathbf{a}\right|^2\right],$$

2. Flow Matching-based Policy provides more powerful modeling for complex distributions by learning a velocity field v_θ that transforms noise ϵ into the expert action $\mathbf{a}$:

$$\mathcal{L}_{\mathrm{F}} = \mathbb{E}_{\tau\sim\mathcal{U}(0,1),\,(o,a)\sim\mathcal{D}}\left[\left\|v_\theta(a^\tau, o, t) - (a - \epsilon)\right\|^2\right],$$

where $\mathbf{x}_\tau = (1-\tau)\boldsymbol{\epsilon} + \tau\mathbf{a}$, and $\tau \sim \mathcal{U}(0,1)$. Common detailed training setups are provided in Appendix D, while specific learning regimes are discussed independently in the subsequent analysis sections. We further introduce a **Foundation Policy** (π_0), designed specifically to investigate the transfer learning properties across different action spaces. Further discussion for action space design with pretraining-finetuning paradigm is provided in Appendix B.

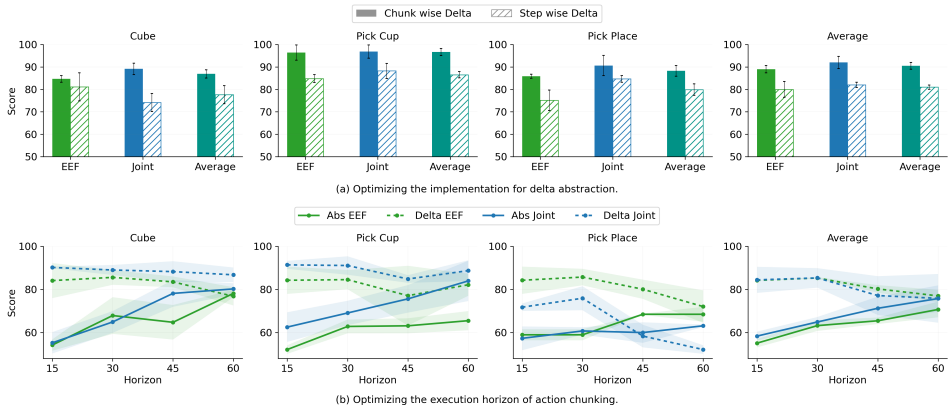


Figure 3: (a) We verified that chunk-wise delta for both EEF and Joint perform better than step-wise delta representations. (b) Grid search over execution horizons across four different action space.

3.2 Robotic Platforms and Evaluation Protocol

Our experiments are conducted across four distinct robotic hardware configurations: (1) a single-arm AgileX platform serving as the primary setup for large-scale real-world experiments; (2) a dual-arm AgileX platform and (3) a single-arm AIRBOT platform, both of which are utilized to evaluate cross-platform generalizability; and (4) RoboTwin 2.0, a simulation benchmark designed for large-scale, reproducible experiments under controlled environments. The overview and detailed description of our experimental setup can be found in Figure 1(b) and Appendix E, respectively.

Evaluation Protocol for Real-World Experiments. For the real-world experiments, we designed a curriculum consisting of four manipulation tasks: `Touch Cube`, `Pick Up Cup`, `Pick and Place Cup`, and `Bimanual Cube Transfer`. These tasks are characterized by increasing contact richness, temporal horizons, and coordination requirements. Detailed task descriptions are provided in Appendix D. To ensure reproducibility and statistical significance, we implement a rigorous evaluation protocol to ensure spatial coverage. Specifically, the robot's workspace is uniformly partitioned into a 6×6 grid. Both data collection and testing procedures strictly adhere to this protocol by initializing object positions uniformly across these grids, thereby mitigating potential distribution shifts between training and evaluation. For each real-world experiment, we report progress score based on three independent trials, where each trial comprises 10 individual rollouts.

Evaluation Protocol for Simulation. We adopt the AgileX embodiment within RoboTwin 2.0 simulation environment and select a subset of 10 tasks from the 50 officially provided tasks. Aligned with our real-world evaluation protocol, we report the average success rate across three independent trials, with each trial consisting of 10 rollouts per task.

4 Results and Analyses

In this section, we systematically evaluate different action space by addressing three progressive research questions (RQs), moving from foundational implementation nuances to generalizable trends and large-scale robustness:

> **RQ1 (Foundational Impact):** *At the implementation level, how do specific choices in action space realization influence policy performance, and what constitutes the optimal configuration?*
>
> **RQ2 (Generalizable Trends):** *Building upon these optimized implementations, can we identify consistent trends across diverse tasks that dictate the selection of superior action abstractions?*
>
> **RQ3 (Systemic Robustness):** *Finally, do the identified trends remain robust when subjected to more advanced settings, such as scaled data regimes, foundation model transfer, and cross-embodiment learning?*

Table 1: Quantitative comparison of progress scores and standard errors across embodiments and tasks. The results contrast Regression (ACT) and Flow Matching (DP) under four distinct control interface configurations. **Bold** and underlined values denote the best and second-best performance for ACT and DP separately.

Task	EE (ACT)		Joint (ACT)		EE (DP)		Joint (DP)	
	abs	delta	abs	delta	abs	delta	abs	delta
Single Arm AgileX								
Cube	77.1 ± 3.8	**86.2** ± 3.5	77.2 ± 2.8	84.8 ± 3.6	83.6 ± 2.0	91.5 ± 3.7	95.5 ± 2.3	**96.7** ± 1.8
Pick	63.1 ± 3.7	**97.9** ± 2.1	83.9 ± 9.8	95.2 ± 4.8	64.6 ± 7.5	**97.9** ± 2.1	77.7 ± 8.6	97.6 ± 2.4
Pick Place	66.8 ± 6.5	**84.7** ± 6.4	70.8 ± 2.8	83.8 ± 4.6	73.8 ± 1.2	84.8 ± 1.9	81.9 ± 4.1	**93.5** ± 0.3
Average	69.0 ± 2.0	**89.6** ± 2.1	77.3 ± 2.8	88.0 ± 2.9	74.0 ± 3.1	91.4 ± 1.6	85.0 ± 2.3	**95.9** ± 1.1
Single Arm AgileX (Multi)								
Cube	89.7 ± 5.7	**96.4** ± 3.6	94.4 ± 3.1	88.8 ± 3.5	96.4 ± 3.6	93.2 ± 4.1	97.9 ± 2.1	**100.0** ± 0.0
Pick	67.6 ± 9.2	89.3 ± 4.0	77.7 ± 6.9	**95.8** ± 4.2	91.1 ± 2.7	**100.0** ± 0.0	90.8 ± 6.4	**100.0** ± 0.0
Pick Place	52.2 ± 10.0	72.3 ± 6.1	**73.8** ± 7.3	72.8 ± 2.8	75.0 ± 5.4	83.5 ± 2.6	86.2 ± 4.3	**93.3** ± 3.4
Average	69.8 ± 1.0	**86.0** ± 3.1	81.9 ± 4.2	85.8 ± 0.9	87.5 ± 2.8	92.2 ± 2.3	91.6 ± 1.1	**97.8** ± 1.1
Bimanual AgileX								
Bowl	63.7 ± 8.3	67.0 ± 5.2	51.6 ± 7.7	**69.6** ± 5.5	64.6 ± 7.5	**75.3** ± 9.3	74.7 ± 3.9	74.6 ± 5.2
RoboTwin 2.0								
Average	26.7 ± 3.3	33.3 ± 6.1	40.0 ± 0.6	**46.3** ± 1.9	26.0 ± 1.2	37.0 ± 6.2	32.3 ± 2.6	**48.0** ± 4.4
Overall Avg	63.4 ± 2.7	78.4 ± 1.4	71.2 ± 2.9	**79.7** ± 2.5	71.9 ± 4.8	82.9 ± 1.6	79.6 ± 2.2	**88.0** ± 2.3

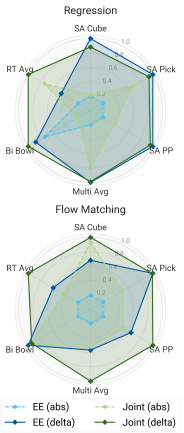


Figure 4: Normalized Score Comparison.

4.1 RQ1: Implementation Nuances are Decisive

We begin by addressing the implementation ambiguities introduced by *action chunking* as identified in Sec. 2.3. These implementation nuances are often overlooked but, as we demonstrate, are decisive for policy stability. We provide more implementation nuances in Appendix F.

4.1.1 Superiority of Chunk- vs. Step-wise Delta

We first conduct real-world experiments to evaluate the performance impact of chunk-wise and step-wise delta actions. Fig. 3(a) reports the performance of a standard regression-based policy on a single-arm AgileX platform across three foundational tasks: `Touch Cube`, `Pick Up Cup`, and `Pick and Place Cup`. Extended cross-validation results and training details are available in Appendix F. Our results demonstrate that ***chunk-wise delta*** **consistently and significantly outperforms step-wise delta across all tasks**. Notably, the performance gap reaches upwards of **10%** on average, a substantial margin that underscores the importance of alignment frame selection.

To explain the underlying mechanism, we analyze the action decoding process under a fundamental stability criterion: ***a robust action representation should not amplify prediction errors during deployment.***

Proposition 4.1 (Noise Amplification in Step-wise Integration)**.** *Let $\boldsymbol{\epsilon} \in \mathbb{R}^k$ be the prediction noise for a chunk of length k, with bounded norm $\|\boldsymbol{\epsilon}\|_2 \leq \delta$. The cumulative error in the decoded executable actions, denoted as $\mathbf{e}_a$, relates to $\boldsymbol{\epsilon}$ via a linear transformation matrix $\mathbf{M}$:*
*(1) For **step-wise delta**, $\mathbf{M}_{\text{step}} = \mathbf{L}_k$, where $\mathbf{L}_k$ is the $k \times k$ lower-triangular matrix of ones. The worst-case error bound scales linearly with the horizon: $\|\mathbf{e}_a\|_2 \leq \|\mathbf{L}_k\|_2 \|\boldsymbol{\epsilon}\|_2 \approx \frac{2k+1}{\pi}\delta \sim \mathcal{O}(k)$.*
*(2) For **chunk-wise delta** and **absolute** action, $\mathbf{M} = \mathbf{I}_k$, implying independent error propagation: $\|\mathbf{e}_a\|_2 \leq \|\mathbf{I}_k\|_2 \|\boldsymbol{\epsilon}\|_2 = \delta \sim \mathcal{O}(1)$.*

Proof and detailed analysis see Appendix G.2. As demonstrated in Proposition 4.1, *step-wise* integration inherently amplifies prediction noise as the horizon k increases, whereas *chunk-wise* and *absolute* action **maintain a constant error bound**. This theoretical result corroborates our empirical findings, confirming that chunk-wise delta yields a structurally more reliable representation.

4.1.2 Interplay With Horizon k

Building upon the optimized chunk-wise implementation, we investigate the chunking horizon k. Following the experimental setup described in Sec. 4.1, we conducted a grid search across horizons. Crucially, following the practice in π Black et al. (2025), all policies were trained using a consistent, longer horizon of $k = 60$ (2 seconds at 30 Hz) to ensure maximum supervision efficiency and temporal coherence. During inference, we then grid-searched the execution horizon from 15 to 60 to identify the optimal deployment window for each representation.

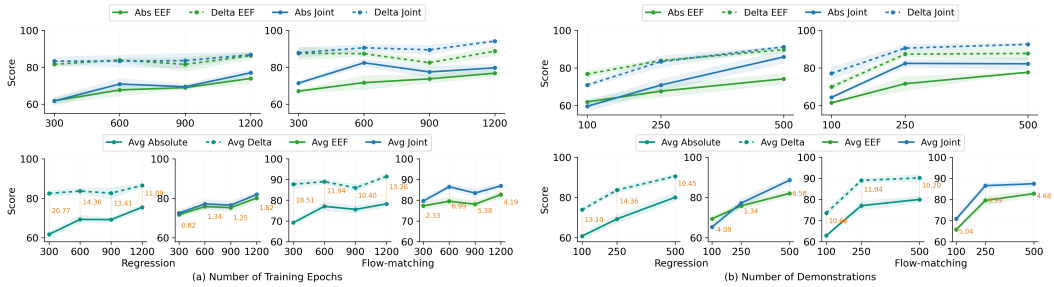


Figure 5: **Consistency of Action Space Superiority under Scaling.** We evaluate policy performance across varying (a) training epochs and (b) number of demonstrations. The top row illustrates the individual performance of the four action space, while the bottom row aggregates the performance across temporal (*abs* vs. *delta*) and spatial (*task* vs. *joint*) dimensions, providing an intuitive comparison.

The results in Fig. 3(b) reveal an insightful phenomenon: **absolute control benefits from a significantly longer horizon, whereas delta control peaks at a shorter horizon**. This observation aligns with our hypothesis in Sec. 2.3 regarding the sensitivity of relative representations to execution drift. Notably, in several tasks, we observe a saturation point even for absolute actions, where increasing the execution horizon no longer yields significant gains. This phenomenon suggests a potential **information decorrelation** effect (See Appendix G). Managing the trade-off between the stability provided by long-horizon absolute grounding and the inherent information decay of distant predictions remains a compelling avenue for future research. We provide further discussion in Appendix B.

Takeaway

Implementation Nuances are Decisive:
(1) *Chunk-wise delta* is fundamentally superior to *step-wise*. (2) Optimal horizons are abstraction-dependent: *delta* requires shorter while *absolute* thrives with longer horizons.

4.2 RQ2: Systematic Trends in Action Abstraction

With the implementation strategies optimized in RQ1, we standardize all delta-based actions to the *chunk-wise* alignment frame. In addition, to accommodate the horizon-coupling effect identified in earlier analysis, we employ shorter execution horizons ($k = 30$) for *delta* actions and longer horizons ($k = 60$) for *absolute* actions for optimal performance. With this foundation, we then pivot to the central inquiry: *how do different action space influence performance across diverse embodiments, model variations, and learning regimes?*

To investigate these dimensions systematically, we conducted extensive experiments across three platforms: a single-arm AgileX robot, a bimanual AgileX robot, and the RoboTwin-2.0 simulation environment. Our evaluation spans 14 distinct tasks in total: 4 real-world tasks and 10 simulation tasks, as described in Sec 3. For data collection, we utilized 250 expert demonstrations per task for real-world experiments and 50 per task for the RoboTwin-2.0. To ensure the generality of our findings, we evaluated both standard regression-based and flow-matching-based policy networks, each trained for 600 epochs. Furthermore, we introduced both single- and multi-task learning settings on the single-arm platform to examine how different abstractions withstand task interference and distribution shifts. Detailed experimental specifications are provided in Appendix E.1.

The comprehensive experimental results are reported in Table 1. Additionally, we provide a radar plot (Fig. 4) with normalized scores to highlight the difference and facilitate an intuitive comparison across different action spaces. In the subsequent sections, we conduct a deep dive into the spatial and temporal abstraction axes, independently analyzing the superiority of various action abstractions.

4.2.1 Temporal Abstraction

We first conduct an in-depth analysis along the **temporal axis**. Even when utilizing the optimal implementation identified for both *absolute* and *delta* abstractions in RQ1, a substantial performance gap remains evident. Specifically, we observe that **with standard modern practice, *delta* abstraction consistently and significantly outperforms *absolute* abstraction across all platforms, task configurations, and model variations.**

We attribute this superiority to two primary factors: (1) As discussed in Sec 2, learning a direct mapping from high-dimensional visual observations to global coordinates presents a significant challenge. Even for policies with strong expressiveness and proper normalization, global coordinates in both *Task* and *Joint* spaces exhibit lower local coherence. In contrast, the properly implemented *chunk-wise delta* actions allow the network to focus on the immediate displacement, which is a more tractable inductive bias. (2) There has been insufficient exploration of the horizon trade-off, as noted in Sec 2.3. While *absolute* representations prefer a longer execution horizon to maintain global consistency, training policies with long horizons remains a complex challenge.

While we hope these findings encourage further exploration into *absolute* representations to leverage their potential for precise global grounding, the deterministic conclusion from our current empirical study is that ***delta* abstraction provides a more robust and sample-efficient foundation for modern imitation learning backbones.**

4.2.2 Spatial Abstraction

Along the axis of spatial abstraction, we observe **an overall performance superiority for actions in *Joint space* compared to *Task space***. However, unlike the deterministic conclusion reached in the temporal analysis, this spatial superiority contains some inconsistency across different platforms, tasks, and learning regimes.

Amidst this complex and entangled data, we identify a particularly insightful phenomenon: **policies trained under the flow-matching generative paradigm exhibit a distinct excellence in *Joint space* learning**. Specifically, as shown in Figure 4, the performance envelope for flow-matching-based models significantly expands when transitioned from *task* to *joint space*. These results are closely aligned with our discussion in Sec 2: as the action distribution in *joint space* often resides on a complex, non-linear, and multi-modal hardware configuration manifold Bensadoun et al. (2022), powerful generative modeling is required to effectively capture the underlying structure. While standard regression backbones often struggle with such complexity and multi-modality, the flow-matching paradigm excels at modeling these intricate joint-space distributions, thereby unlocking the full potential of intrinsic control stability and preserving the essential kinematic meaning of *Joint space*.

Takeaway

(1) ***Delta actions*** serve as a superior temporal abstraction for modern policy backbones. (2) ***Joint-space control*** generally provides a more robust spatial representation, particularly when paired with **strong generative modeling** (e.g., diffusion).

4.3 RQ3: Consistency and Scaling Analysis

In this section, we extend our experiments to broader settings and larger-scale validations under varied control conditions. We investigate whether the conclusions derived in RQ2 hold firm when the learning regime is scaled in terms of data volume and computation budgets. Furthermore, we evaluate these abstractions within advanced learning setups, including transfer learning from pretrained robotics foundation models and cross-embodiment learning scenarios.

4.3.1 Scaling with Data and Compute

We extended the RQ2 experiments to a much larger scale, covering all three hardware platforms, learning regimes, and model architectures. To evaluate *Compute Scaling*, we introduced three additional training milestones at 600, 900, and 1200 epochs. For *Data Scaling*, we varied the demonstration volume (100, 250, and 500 trajectories) for real-world experiments. The results of single-task learning on the AgileX platform across the four designed real-world tasks, aggregated in Figure 5, provide a high-fidelity map of action abstraction performance across varying resource budgets. Each data point in the figure represents the averaged outcome of 12 independent validation trials across different setups (totaling 120 real-world rollouts per point). Further exhaustive cross-validation is presented in Appendix F.

Crucially, these results substantiate a definitive trend: *delta actions* consistently serve as the superior temporal abstraction, albeit with marginal gains in certain settings, while *joint-space actions* remain

the most robust spatial representation in most cases. Notably, as training epochs and data volume increase, the superiority of *joint-space* actions becomes increasingly pronounced, particularly for regression-based policies. This mirrors our observations in RQ2: while *task-space* can be competitive in low-data or limited-compute regimes, *joint-space* benefits disproportionately from stronger modeling capabilities and extensive training. This suggests that *joint-space* representations are **fundamentally better suited to capturing the underlying kinematic manifold as the learning regime scales**.

4.3.2 Advanced Learning Regimes

To align with increasing learning demands and the objective of pursuing a more generalizable embodied agent, this section examines two prominent learning regimes: cross-embodiment learning and transfer learning from foundation models. Specifically, we introduce a new embodiment, AIRBOT, for cross-embodiment learning and employ π_0 as the foundation model to perform transfer learning. We report the average cross-embodiment performance across the AgileX and AIRBOT platforms in Figure 6(a), and the transfer learning results across three single-arm tasks on AgileX in Figure 6(b). See Appendix E.3 for experimental details.

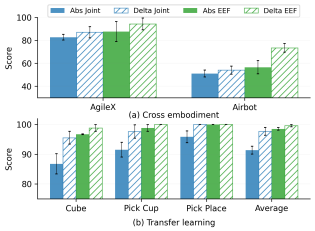


Figure 6: Comparison across different action space on advanced learning settings

The results again confirm the superiority of *delta control* compared to *absolute actions*. However, regarding spatial representations, we observe an interesting shift: under cross-embodiment and transfer learning settings, *task-space* representations exhibit a more pronounced advantage and, in some cases, surpass *joint-space* control. We attribute this behavior to the relatively embodiment-invariant nature of *task-space* representations, which abstract away robot-specific kinematics and thereby facilitate knowledge transfer across different embodiments. These results highlight a complementary strength of *task-space* control in scenarios where generalization across robots or tasks is prioritized over execution robustness within a fixed embodiment.

Takeaway

(1) The superiority of ***delta actions*** remains consistent across diverse learning regimes. (2) ***joint-space actions*** benefit exceptionally from stronger modeling capacity and extensive training. (3) ***task-space representations*** demonstrate a pronounced advantage in generalized settings, specifically within cross-embodiment and transfer learning regimes.

5 Conclusion and Practical Implications

In this work, we presented a systematic evaluation of action space design for IL-based robotic manipulation, decomposing the problem into orthogonal spatial and temporal axes. By conducting large-scale experiments across 13,000+ real-world rollouts and extensive testing in simulation environment, we establish that action space is far from a trivial implementation detail. Instead, our results reveal that action representation is a decisive configuration that interacts non-trivially with diverse learning regimes. We summarize our findings into the following actionable guidelines to standardize future research and deployment:

1. The execution horizon k of action chunking should not be treated as an isolated constant but must be adapted to the temporal abstraction.
2. For standard imitation learning settings with sufficient resources where the primary objective is to maximize performance on a specific hardware platform (e.g., single-arm manipulation), the combination of ***joint space*** and ***chunk-wise delta*** yields the most robust results.
3. When the objective shifts towards generalized setting like cross-embodiment or transfer learning, ***task space*** **(EE)** becomes the superior spatial abstraction.

Despite the unprecedented scale of this study, our derived insights represent a first step towards a unified understanding of action spaces. We highlight several exciting directions that remain to be explored in Appendix B, paving the way for future research.

REFERENCES

Raphael Bensadoun, Shir Gur, Nitsan Blau, Tom Shenkar, and Lior Wolf. Neural inverse kinematics, 2022. URL https://arxiv.org/abs/2205.10837.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, brian ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *9th Annual Conference on Robot Learning*, 2025.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023a.

Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on robot learning*, pp. 287–318. PMLR, 2023b.

Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

Chilam Cheang, Sijin Chen, Zhongren Cui, Yingdong Hu, Liqun Huang, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Xiao Ma, et al. Gr-3 technical report. *arXiv preprint arXiv:2507.15493*, 2025.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Qiwei Liang, Zixuan Li, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, pp. 02783649241273668, 2023.

Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. Robotics: Science and Systems, 2024.

Ria Doshi, Homer Walke, Oier Mees, Sudeep Dasari, and Sergey Levine. Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation. *arXiv preprint arXiv:2408.11812*, 2024.

Yilun Du, Sherry Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B Tenenbaum, Leslie Pack Kaelbling, et al. Video language planning. In *The Twelfth International Conference on Learning Representations*.

Julian Eßer, Gabriel B Margolis, Oliver Urbann, Sören Kerner, and Pulkit Agrawal. Action space design in reinforcement learning for robot motor skills. In *8th Annual Conference on Robot Learning*, 2024.

Huang Fang, Mengxi Zhang, Heng Dong, Wei Li, Zixuan Wang, Qifeng Zhang, Xueyun Tian, Yucheng Hu, and Hang Li. Robix: A unified model for robot interaction, reasoning and planning. *arXiv preprint arXiv:2509.01106*, 2025.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pp. 991–1002. PMLR, 2022.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

CS George Lee. Robot arm kinematics, dynamics, and control. *Computer*, 15(12):62–80, 1982.

Seungjae Lee, Yibin Wang, Haritheja Etukuru, H. Jin Kim, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Behavior generation with latent actions. In *Forty-first International Conference on Machine Learning*, 2024.

Chengmeng Li, Junjie Wen, Yan Peng, Yaxin Peng, Feifei Feng, and Yichen Zhu. Pointvla: Injecting the 3d world into vision-language-action models. *arXiv preprint arXiv:2503.07511*, 2025a.

Jianxiong Li, Jinliang Zheng, Yinan Zheng, Liyuan Mao, Xiao Hu, Sijie Cheng, Haoyi Niu, Jihao Liu, Yu Liu, Jingjing Liu, Yaqin Zhang, and Xianyuan Zhan. Decisionnce: Embodied multimodal representations via implicit preference learning. In *Forty-first International Conference on Machine Learning*.

Zezeng Li, Alexandre Chapin, Enda Xiang, Rui Yang, Bruno Machado, Na Lei, Emmanuel Dellandrea, Di Huang, and Liming Chen. Robotic manipulation via imitation learning: Taxonomy, evolution, benchmark, and challenges, 2025b. URL `https://arxiv.org/abs/2508.17449`.

Fanqi Lin, Yingdong Hu, Pingyue Sheng, Chuan Wen, Jiacheng You, and Yang Gao. Data scaling laws in imitation learning for robotic manipulation. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL `https://openreview.net/forum?id=pISLZG7ktL`.

Fanqi Lin, Ruiqian Nai, Yingdong Hu, Jiacheng You, Junming Zhao, and Yang Gao. Onetwovla: A unified vision-language-action model with adaptive reasoning. *arXiv preprint arXiv:2505.11917*, 2025b.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Dongxiu Liu, Haoyi Niu, Zhihao Wang, Jinliang Zheng, Yinan Zheng, Zhonghong Ou, Jianming HU, Jianxiong Li, and Xianyuan Zhan. Efficient robotic policy learning via latent space backward planning. In *Forty-second International Conference on Machine Learning*.

Dongxiu Liu, Haoyi Niu, Zhihao Wang, Jinliang Zheng, Yinan Zheng, Zhonghong Ou, Jianming Hu, Jianxiong Li, and Xianyuan Zhan. Efficient robotic policy learning via latent space backward planning, 2025a. URL `https://arxiv.org/abs/2505.06861`.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1b: a diffusion foundation model for bimanual manipulation. In *The Thirteenth International Conference on Learning Representations*, 2025b.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

NVIDIA, :, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025.

Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. π0.5: a vision-language-action model with open-world generalization. *URL https://arxiv. org/abs/2504.16054*, 1(2):3, 2025.

Lucy Xiaoyang Shi, Michael Robert Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, Adrian Li-Bell, et al. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. In *Forty-second International Conference on Machine Learning*.

Gordon D Smith. *Numerical solution of partial differential equations: finite difference methods*. Oxford university press, 1985.

Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023.

Zhengrong Xue, Shuying Deng, Zhenyang Chen, Yixuan Wang, Zhecheng Yuan, and Huazhe Xu. Demogen: Synthetic demonstration generation for data-efficient visuomotor policy learning. In *Synthetic Data for Computer Vision Workshop@ CVPR 2025*.

Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, et al. Latent action pretraining from videos. *arXiv preprint arXiv:2410.11758*, 2024.

Jiawen Yu, Hairuo Liu, Qiaojun Yu, Jieji Ren, Ce Hao, Haitong Ding, Guangyu Huang, Guofan Huang, Yan Song, Panpan Cai, Wenqiang Zhang, and Cewu Lu. ForceVLA: Enhancing VLA models with a force-aware moe for contact-rich manipulation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=2845H8Ua5D`.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024.

Thomas T. Zhang, Daniel Pfrommer, Chaoyi Pan, Nikolai Matni, and Max Simchowitz. Action chunking and exploratory data collection yield exponential improvements in behavior cloning for continuous control, 2025a. URL `https://arxiv.org/abs/2507.09061`.

Zongzheng Zhang, Haobo Xu, Zhuo Yang, Chenghao Yue, Zehao Lin, Huan-ang Gao, Ziwei Wang, and Hao Zhao. Ta-vla: Elucidating the design space of torque-aware vision-language-action models. *arXiv preprint arXiv:2509.07962*, 2025b.

Tony Z Zhao, Jonathan Tompson, Danny Driess, Pete Florence, Seyed Kamyar Seyed Ghasemipour, Chelsea Finn, and Ayzaan Wahid. Aloha unleashed: A simple recipe for robot dexterity. In *8th Annual Conference on Robot Learning*.

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

Jinliang Zheng, Jianxiong Li, Sijie Cheng, Yinan Zheng, Jiaming Li, Jihao Liu, Yu Liu, Jingjing Liu, and Xianyuan Zhan. Instruction-guided visual masking. *Advances in neural information processing systems*, 2024.

Jinliang Zheng, Jianxiong Li, Dongxiu Liu, Yinan Zheng, Zhihao Wang, Zhonghong Ou, Yu Liu, Jingjing Liu, Ya-Qin Zhang, and Xianyuan Zhan. Universal actions for enhanced embodied foundation models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 22508–22519, 2025.

A Ethics and Reproducibility Statement

In this paper, we employed Large Language Models (LLMs) **solely** for polishing the writing. No parts of the technical content, experimental results, or conclusions were generated by LLMs. All real-world data utilized in this study were collected using our custom hardware platform following the protocols detailed in the paper. Simulation data were generated using open-source environments. To support reproducibility, all code and datasets will be **open-sourced** upon publication. We confirm that our datasets contain no personally identifiable information and pose no risks regarding privacy violations or social bias.

B Limitations and Future Work

In this section, we look beyond the scope of our current benchmarking to identify several prospective directions for action space design. While our study establishes a strong baseline, we believe the following areas hold significant promise for unlocking the next generation of robotic control.

1. Beyond Rigid Taxonomies: Hybrid and Adaptive Representations. Our current analysis operates within a fixed taxonomy (e.g., strictly *Absolute* vs. *Delta*, or *Joint* vs. *Task*). However, the optimal action space may not be static. A compelling avenue for future work is to explore **hybrid or adaptive action spaces** that dynamically switch representations based on the task phase. For instance, a policy might utilize *task-space delta* actions for the reaching phase to maximize generalization, while seamlessly transitioning to *joint-space absolute* actions for fine-grained manipulation or contact-rich stages. Learning to modulate these representations end-to-end could theoretically combine the stability of global grounding with the local precision of relative control. Another significant but under-explored dimension is the theoretical underpinning of *Action Chunking*. While Sec. 2.3 addresses the practical implementation nuances, our understanding of this mechanism remains empirical. Current strategies for selecting chunking horizons are heuristic and likely suboptimal. For instance, while our results demonstrate that extending the training horizon benefits *absolute actions*, training with longer horizons introduces distinct, under-explored challenges regarding convergence stability and information decorrelation. A critical gap remains in rigorously formalizing **how action chunking reshapes the optimization landscape** to derive principled, rather than heuristic, horizon selection strategies.

2. Scaling to High-DoF Morphologies, Dynamic and Dexterous Tasks. While our experiments cover several standard platforms for robotic manipulation, the intrinsic connection between action space and **morphological complexity** warrants deeper investigation. It remains an open question whether our findings—specifically the superiority of delta joint actions—generalize to high-degree-of-freedom systems, such as humanoids or multi-fingered hands, where the kinematic manifold exhibits significantly higher dimensionality and non-linearity. Furthermore, extending this evaluation to highly dynamic or dexterous domains, such as table tennis or deformable object manipulation (e.g., cloth folding), could uncover critical constraints on action latency and horizon coupling that are less visible in pick-and-place tasks.

3. Unifying Action Spaces for Generalization and Transfer. In Sec. 4.3, we conduct a preliminary investigation into the interplay between action space design and cross-embodiment generalization. Our results, spanning cross-embodiment learning across distinct morphologies and transfer from the π_0 foundation model, demonstrate a **pronounced superiority** of *task space (EE)* control. This advantage is primarily attributable to its inherent embodiment invariance. Nevertheless, this observation requires broader validation. A critical next step is to extend the analysis to a wider range of embodiment pairs and foundation models, with particular attention to how different pretraining paradigms, such as vision language alignment and pure behavioral cloning, interact with action space selection. More specifically, an important open question is whether aligning pretraining supervision directly in *joint space* can reduce the transfer gap and potentially challenge the current dominance of task-space representations in foundation models.

C Related Work

Learning-Based Robotic Manipulation Policies. Robotic manipulation policies have made remarkable progress in recent years, advancing from simple atomic pick-and-place tasks Kim et al.

(2024); Brohan et al. (2022; 2023a); O'Neill et al. (2023), to long-horizon sequential tasks Du et al.; Brohan et al. (2023b); Fang et al. (2025); Shi et al., fine-grained contact-rich manipulations Chi et al. (2023); Yu et al. (2025); Zhang et al. (2025b), and even complex dexterous skills Zhao et al.; Physical Intelligence et al. (2025); Bjorck et al. (2025); Team et al. (2025); Zhao et al. (2023). These policies are typically formulated as end-to-end models: given images, robot proprioceptive states, and optionally language prompts or other modalities such as point clouds Li et al. (2025a); Xue et al.; Ze et al. (2024) or force feedback Zhang et al. (2025b); Yu et al. (2025), the model must generate *actions* that directly control the robot. Auxiliary supervision signals, such as future image prediction Cheang et al. (2024); Bjorck et al. (2025); Cheang et al. (2025); Ye et al. (2024); Li et al.; Wen et al. (2023); Liu et al., object detection Zheng et al. (2024); Team et al. (2025), or sub-language planning Black et al. (2025); Shi et al.; Lin et al. (2025b), are introduced to leverage web knowledge and improve generalization. However, *ACTION* is the only modality that can be the interface for robotics models to interact with the 3D world, and thus is the indispensable modality for robotic learning that ultimately governs execution performance, making it one of the most critical design choices in policy learning Lee et al. (2024); Chi et al. (2023).

Chaotic Control Interfaces for Robotic Policies. The physical representations of actions vary widely, providing numerous options for supervision. A fundamental choice is whether to represent actions in joint space that directly control motors, or in EEF space, which specifies the gripper's position and orientation in 3D space Doshi et al. (2024). At first glance, these two representations may seem interchangeable, since forward kinematics (FK) maps joints to EEF poses and inverse kinematics (IK) provides the reverse mapping. However, as we show in this paper, joint-based and EEF-based actions exhibit markedly different training behaviors and preferences. Furthermore, actions can be parameterized in different orders, such as 0th-order positions Liu et al. (2025b); Chi et al. (2023); Zhao et al. (2023), 1st-order velocities Team et al. (2024); Doshi et al. (2024), or even higher-order derivatives, adding further variability. To date, no clear consensus has emerged on which representation is most effective, where most different works adopt different interfaces without concrete reasons Chi et al. (2023); Zhao et al. (2023); Physical Intelligence et al. (2025); Doshi et al. (2024); Chi et al. (2024).

D Model Implementation and Training Details

In this section, we provide more details about the model implementation and training procedure. The overview of our implemented model architecture is illustrated in Fig 7. Following prior works Liu et al. (2025a); Zhao et al. (2023); Chi et al. (2023), our model comprises a **FiLM-conditioned ResNet-18** backbone that injects language features into visual representations, and a **Transformer-based encoder-decoder** for action generation. Our architecture supports two prominent training paradigms as described in Sec 3: (1) **Regression** trained with L2 loss for direct action prediction, and (2) a **Flow-Matching method** that predicts the target vector field for generative modeling.

Regarding the training setup, unless otherwise specified, we follow the hyperparameters listed in Table 2 to **train all models**. All training procedures are conducted using 8 NVIDIA A100 GPUs, **resulting in an overall computational cost exceeding 16,000 GPU-hours.**

Configuration	Value
Optimizer	AdamW
Batch size	512
Learning rate	1×10^{-4}
LR Scheduler	CosineAnnealingLR
Weight decay	0.01
Optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.95$
Model precision	float32
Image Resize	224x224
Image Augmentation	ColorJitter(0.2, 0.2, 0.2, 0)

Table 2: Hyperparameters for model training.

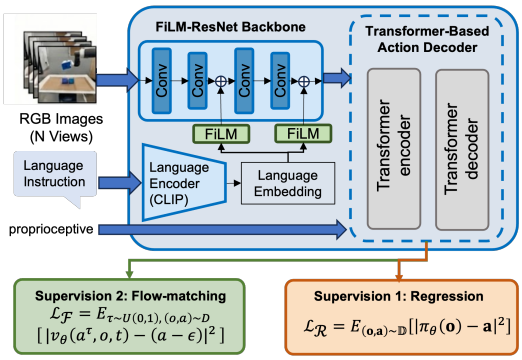


Figure 7: Overview of Model architecture.

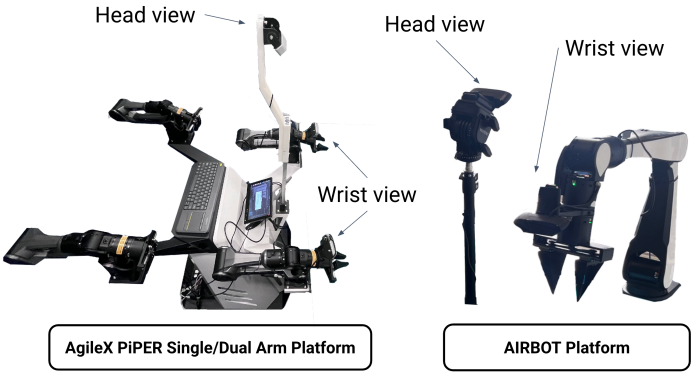


Figure 8: Overview of hardware setup for real world experiments

Table 3: **Task Curriculum, Complexity Progression, and Success Criteria.** We design four tasks with increasing difficulty to stress different failure modes of action representations and define rigorous evaluation metrics for each.

Task	Goal	Progress Score	Evaluation Focus
Touch Cube	**Sanity Check**: Move from a random pose to touch a fixed cube.	0.5 for aiming the correct direction; 0.75 for touching the cube corner; 1.0 for touching the top surface.	Isolates *spatial precision* from dynamics; validates basic convergence.
Pick Cup	**Grasping**: Grasp and lift a target cup to a height of 10 cm.	0.5 for touching the cup; 1.0 for picking up the cup.	Introduces *contact dynamics* and gripper timing coordination.
Pick & Place	**Sequential**: Grasp a cup and place it onto a target plate.	0.5 for touching the cup; 0.75 for picking up the cup but failing to put it on the plate; 1.00 for successfully placing it on the plate.	Sensitive to *temporal drift* and error accumulation over a long horizon.
Bimanual Transfer	**Coordination**: Transfer a cube from the left arm to a bowl in the right arm.	0.25 for touching either the cube or the bowl; 0.5 for pick up either; 0.75 for picking up both but failing to put the cube into the bowl; 1.0 for aiming correctly.	Requires *inter-arm coordination* and precise relative positioning.

E Details of Experimental Setup

E.1 Real-World Experiments

In this section, we provide a detailed description of the hardware configurations and task suite used to evaluate the performance of different action spaces in real-world settings. Our empirical study is conducted on three distinct hardware platforms across four tasks, as summarized in Table 3. Visualizations of the hardware setups are provided in Fig. 8.

Single-Arm AgileX PiPER: The single-arm PIPER is a lightweight 6-DoF robotic manipulator, which we use as a baseline platform for single-handed manipulation tasks. The setup is equipped with a third-person camera and a wrist-mounted camera to capture global visual context and fine-grained local information, respectively. This platform evaluates the model's ability to execute precise and dynamic actions, including *Touch Cube*, *Pick Up Cup*, and *Pick and Place Cup*, without the added complexity of bimanual coordination.

Dual-Arm AgileX PiPER Platform: Built upon the AgileX robotics platform, this setup is designed to assess fine-grained control in a bimanual manipulation setting. In addition to a third-person camera, each arm is equipped with a wrist-mounted camera, enabling active perception and coordinated two-arm behaviors. This platform targets contact-rich scenarios that require accurate timing and cross-arm coordination, exemplified by the *Bimanual Transfer* task.

AIRBOT: AIRBOT is a 6-DoF robotic arm characterized by its cost-effectiveness and kinematic structure that differs from the PIPER arms. We include this platform to evaluate the cross-morphology generalization capability of different action spaces on the *Touch Cube* task.

E.2 Simulations

We use RoboTwin2 as the simulation platform to evaluate the performance of different action spaces under an idealized hardware configuration. Specifically, we adopt the AgileX embodiment provided by RoboTwin 2.0 and select 10 representative manipulation tasks: *Adjust Bottle*, *Dump Bin Bigbin*, *Grab Roller*, *Lift Pot*, *Move Playingcard Away*, *Open Laptop*, *Place Burger Fries*, *Place Container Plate*, *Press Stapler*, and *Shake Bottle Horizontally*. **The task set is chosen based on preliminary experiments, excluding tasks that are either saturated or excessively difficult, in order to provide a balanced evaluation across varying levels of complexity.**

E.3 Transfer Learning with π_0

To evaluate transfer capabilities, we fine-tune the pre-trained π_0 policy using Low-Rank Adaptation (LoRA) utilizing the official codebase. The experimental suite includes three tasks: *Touch Cube*, *Pick Cup*, and *Pick & Place*. Demonstrations from all tasks are merged into a unified dataset and sampled jointly during training, performing a multi-task learning paradigm. We conduct experiments across all four combinations of spatial and temporal abstractions to analyze their distinct effects. Training is performed for 30,000 steps with a batch size of 32, with checkpoints saved and validated every 10,000 steps. All other hyperparameters align with the settings described in Table 2.

F Cross Validation

We conduct extensive experiments to substantiate the claims made in this work. While the main text reports the most representative results, this section presents a more comprehensive set of additional experimental evidence to cross-validate our findings and provide further support for our conclusions.

F.1 Cross-Validation of Chunk-Wise vs. Step-Wise Delta Actions

In addition to the regression-based results reported in Sec. 4.1, we further evaluate the *step-wise delta* interface on the *Cube*, *Cup*, and *Pick and Place* tasks, and compare its performance against the *chunk-wise delta* interface using a flow-matching-based backbone. As shown in Fig. 9, both chunk-wise delta end-effector and joint-space representations consistently outperform their step-wise delta counterparts.

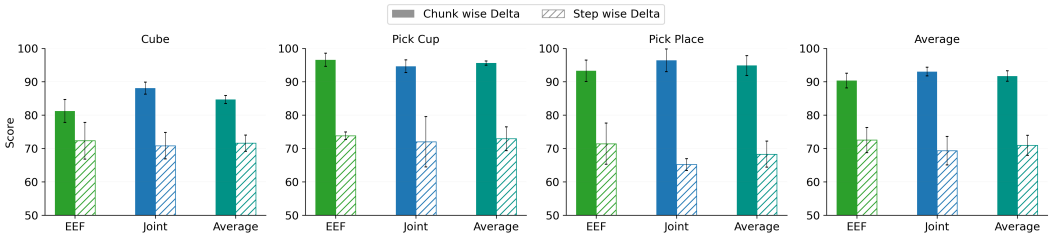
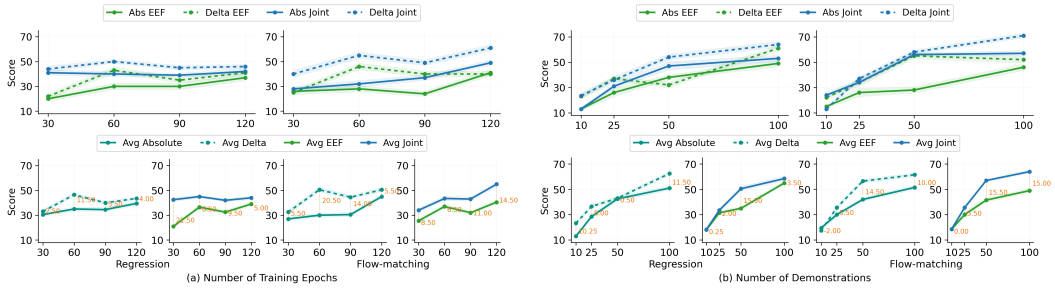


Figure 9: Cross-validation with a flow-matching backbone. Chunk-wise delta representations, in both end-effector and joint space, consistently outperform step-wise delta action interfaces.



(a) Performance vs. Training Epochs

(b) Performance vs. Data Scale

Figure 10: **Results on RoboTwin 2.0 cross-validate the scaling laws of action abstractions.** Comparison of policy performance across (a) training epochs and (b) demonstration scales for regression-based and flow-matching-based backbones. These results confirm that the proposed method scales effectively with both increased compute and data.

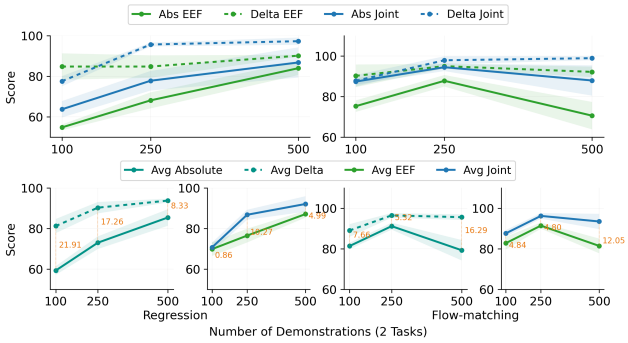


Figure 11: Scaling experiments on multi-task learning

F.2 Simulation Validation: Consistency across Data and Compute Scaling

To verify that our observations regarding action abstractions are not artifacts of specific real-world hardware setups, we conducted large-scale consistency checks in simulation using the RoboTwin 2.0 benchmark. Simulation allows us to rigorously test the scaling laws of our method with significantly larger datasets and more extensive training horizons than are feasible in physical experiments.

As illustrated in Fig. 10, we observe trends that are highly consistent with our real-world findings:

- **Temporal Abstraction:** *delta* actions consistently dominate *absolute* actions across all data volumes and training epochs, confirming that relative motion control provides a more stable learning signal regardless of the regime.
- **Spatial Abstraction:** Similar to the real-world results, *joint-space* representations exhibit superior scaling properties. While Task-space control remains competitive in low-data regimes, Joint-space performance improves significantly as data volume increases, eventually outperforming Task-space by a clear margin.

These simulation results provide strong evidence that our conclusions on action space selection are fundamental to the learning dynamics of embodied agents, rather than being specific to a particular robot morphology or physical environment.

F.3 Cross-Validation in Multi-Task Settings

Beyond single-task mastery, we further validate our approach in a multi-task learning (MTL) setting to ensure that our findings are robust to task interference and varying task distributions. We train a single, unified policy co-conditioned on multiple distinct manipulation tasks to assess how well

different action abstractions handle the increased complexity across different data volumes and training epochs. The results, reported in Fig. 11, indicate that the established trends remain robust.

G Formal Definition and Discussion on Action Space Design

We address the problem of imitation learning for robotic manipulation, formally modeled as learning a policy π_θ that maps observations $\mathbf{o}_t$ to low-level deployable joint commands $\mathbf{u}_t \in \mathbb{R}^{d_q}$ at timestep t.

The policy produces a latent sequence $\mathbf{Z}_t \in \mathbb{R}^{c\times d_a}$, which is subsequently transformed through a two-stage process: *temporal decoding* into an executable sequence $\tilde{\mathbf{A}}_t \in \mathbb{R}^{c\times d_a}$, followed by *spatial projection* into the robot's execution space. An overview of this pipeline is provided in Fig. 12. Here, d_q and d_a denote the dimensions of joint-space commands and action representations, respectively, and $c \in \mathbb{N}$ denotes the chunk length. Prior work has both theoretically and empirically demonstrated that $c > 1$ is critical for effective robotic learning.

G.1 Formalization of Action Space Design

Temporal Decoding Let $\mathbf{Z}_t = [\mathbf{z}_{t,1}, \dots, \mathbf{z}_{t,c}]^\top \in \mathbb{R}^{c\times d_a}$ denote the sequence of latent codes. These latents are decoded into an action sequence $\tilde{\mathbf{A}}_t$ using either a *zeroth-order* (absolute) or *first-order* (incremental) parameterization. In the zeroth-order case, each latent directly specifies the corresponding action. In the first-order case, actions are defined relative to a reference state $\mathbf{s}_t^{\text{ref}} \in \mathbb{R}^{d_a}$:

$$\tilde{\mathbf{a}}_{t+k} = \begin{cases} \mathbf{s}_t^{\text{ref}} + \mathbf{z}_{t,k}, & \text{Chunk}, \\ \mathbf{s}_t^{\text{ref}} + \displaystyle\sum_{j=1}^{k} \mathbf{z}_{t,j}, & \text{Step}. \end{cases} \tag{1}$$

This decoding induces a linear temporal operator $\mathbf{M}_{\text{time}}$ acting on the stacked latent sequence, whose structure is determined by the choice of temporal parameterization.

Spatial Mapping Each decoded action $\tilde{\mathbf{a}}_{t+k}$ is projected into joint space via a spatial operator that optionally depends on the current joint configuration $\mathbf{q}_t \in \mathbb{R}^{d_q}$. We define $\Phi_{\text{IK}} : \mathbb{R}^{d_a} \times \mathbb{R}^{d_q} \to \mathbb{R}^{d_q}$,which implements inverse kinematics mapping task-space targets to joint-space commands. The resulting joint command is

$$\mathbf{u}_{t+k} = \begin{cases} \tilde{\mathbf{a}}_{t+k}, & \text{joint-space}, \\ \Phi_{\text{IK}}(\tilde{\mathbf{a}}_{t+k}, \mathbf{q}_t), & \text{task-space}. \end{cases} \tag{2}$$

This formulation allows the same temporal decoder to pair naturally with either joint or task-space action parameterizations, with Φ_{IK} providing the necessary projection into executable joint commands.

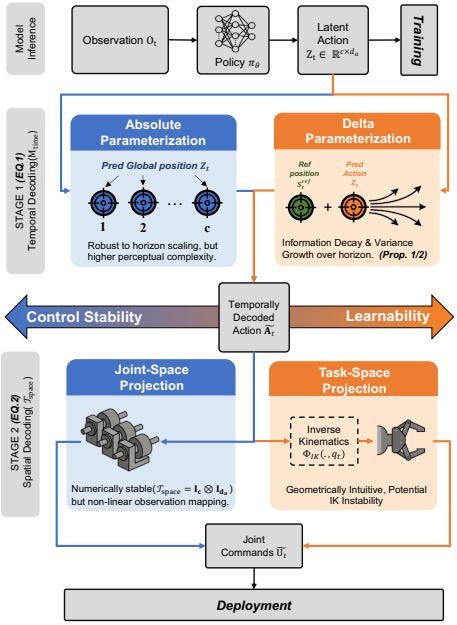


Figure 12: Problem reformulation.

Combined Linear Approximation and Structural Instability To understand how temporal and spatial parameterizations jointly affect execution stability, we analyze the local sensitivity of the full action transformation. Small perturbations in the latent sequence should not produce disproportionately large deviations in the executed commands. To make this dependence explicit, we linearize the composite mapping from latent codes $\mathbf{Z}_t$ to joint-space actions. Let $\mathcal{T}_{\text{space}}$ denote the spatial projection and define its local Jacobian

$$\mathbf{S}_t = \left.\frac{\partial \mathcal{T}_{\text{space}}}{\partial \tilde{\mathbf{a}}}\right|_{\tilde{\mathbf{a}}=\tilde{\mathbf{a}}_t} \in \mathbb{R}^{d_q\times d_a}. \tag{3}$$

For joint-space parameterizations $\mathbf{S}_t = \mathbf{I}_{d_q}$; for task-space mappings, $\mathbf{S}_t$ may represent a differential IK Jacobian or any differentiable projection. Stacking across the horizon yields the block-diagonal operator $\mathbf{S}_{\text{space}} = \mathbf{I}_k \otimes \mathbf{S}_t$. The resulting first-order approximation of the full transformation is

$$\mathcal{T}_{\text{total}} \;\approx\; (\mathbf{I}_k \otimes \mathbf{S}_t)\, \mathbf{M}_{\text{time}}. \tag{4}$$

This expression shows that temporal and spatial representations contribute multiplicatively to overall stability: the sensitivity is governed jointly by the spectral properties of the spatial factor $\mathbf{S}_t$ and the temporal operator $\mathbf{M}_{\text{time}}$.

G.2 Research Question on Temporal Reparameterization

Temporal reparameterization determines how errors accumulate over the horizon, how rapidly information decays, and how difficult the resulting mapping is for a policy to learn from a single observation. To understand these trade-offs, we analyze the behavior of different temporal parameterizations below. Here, we recall the Proposition 4.1, and given the detailed proof.

Proposition G.1 (Noise Amplification in Step-wise Integration)**.** *The step-wise delta representation corresponds to the linear temporal operator $\mathbf{M}_{\text{cum}} = \mathbf{L}_k \otimes \mathbf{I}_{d_a}$, where $\mathbf{L}_k \in \mathbb{R}^{k\times k}$ is the lower-triangular cumulative-sum matrix (with ones on and below the diagonal). The spectral norm of this operator satisfies:*

$$\|\mathbf{M}_{\text{cum}}\|_2 = \sigma_{\max}(\mathbf{L}_k) \approx \frac{2k+1}{\pi}.$$

This norm grows linearly with c and strictly exceeds 1 for all $c \geq 2$.

Consequently, step-wise integration necessarily amplifies input prediction noise as the horizon length increases, inducing structural instability.

Proof. We seek to bound the spectral norm $\|\mathbf{M}_{\text{cum}}\|_2$, where $\mathbf{M}_{\text{cum}} = \mathbf{L}_k$ is the $k \times k$ lower-triangular matrix of ones. Recall that the spectral norm is given by the largest singular value: $\|\mathbf{L}_k\|_2 = \sigma_{\max}(\mathbf{L}_k)$.

Directly computing the eigenvalues of $\mathbf{L}_k\mathbf{L}_k^T$ is complex. Instead, we analyze the inverse operator. The inverse of the cumulative sum matrix is the discrete difference operator, denoted as $\mathbf{D}_k = \mathbf{L}_k^{-1}$:

$$\mathbf{D}_k = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ -1 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & -1 & 1 \end{pmatrix}.$$

Using the property of singular values, $\sigma_{\max}(\mathbf{L}_k) = \frac{1}{\sigma_{\min}(\mathbf{D}_k)}$. We compute the eigenvalues of the symmetric matrix $\mathbf{A} = \mathbf{D}_k\mathbf{D}_k^T$. The matrix $\mathbf{A}$ takes the form of a tridiagonal matrix (closely related to the discrete Laplacian):

$$\mathbf{A} = \begin{pmatrix} 1 & -1 & 0 & \cdots \\ -1 & 2 & -1 & \cdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & -1 & 2 \end{pmatrix}.$$

The eigenvalues λ_i of this specific tridiagonal matrix are well-known in numerical analysis Smith (1985):

$$\lambda_i = 2 - 2\cos\left(\frac{(2i-1)\pi}{2k+1}\right) = 4\sin^2\left(\frac{(2i-1)\pi}{2(2k+1)}\right), \quad i = 1, \ldots, k.$$

The singular values of $\mathbf{D}_k$ are the square roots of these eigenvalues: $\sigma_i(\mathbf{D}_k) = 2\sin\left(\frac{(2i-1)\pi}{2(2k+1)}\right)$. The minimum singular value corresponds to $i = 1$:

$$\sigma_{\min}(\mathbf{D}_k) = 2\sin\left(\frac{\pi}{2(2k+1)}\right).$$

Therefore, the spectral norm of the cumulative sum matrix is:

$$\|\mathbf{L}_k\|_2 = \frac{1}{\sigma_{\min}(\mathbf{D}_k)} = \frac{1}{2\sin\left(\frac{\pi}{4k+2}\right)}.$$

Using the Taylor expansion $\sin(x) \approx x$ for small x (valid as k grows):

$$\|\mathbf{L}_k\|_2 \approx \frac{1}{2 \cdot \frac{\pi}{4k+2}} = \frac{2k+1}{\pi}.$$

This confirms that the error amplification factor grows linearly with the chunk size k, proving the proposition. □

Remark on Temporal Decorrelation for Long-Horizon Action Although the Chunk-Delta and Absolute formulation $\tilde{\mathbf{a}}_{t+k} = \mathbf{s}_t + \mathbf{z}_{t,k}$ avoids numerical drift from cumulative integration, it remains fundamentally *open-loop*: each offset $\mathbf{z}_{t,k}$ must be predicted from the single observation $\mathbf{o}_t$ without intermediate feedback. Let the regression or generative target be the displacement $\Delta\mathbf{a}_k^* = \mathbf{a}_{t+k}^* - \mathbf{s}_t$. Any model can only approximate its conditional distribution $p(\Delta\mathbf{a}_k^* \mid \mathbf{o}_t)$, whose inherent uncertainty is characterized by the conditional entropy $H(\Delta\mathbf{a}_k^* \mid \mathbf{o}_t)$. As the prediction horizon increases, two mechanisms enlarge this uncertainty:

1. **Variance Growth.** The displacement magnitude $\|\Delta\mathbf{a}_k^*\|$ generally increases with k, expanding the support of the target distribution and increasing its entropy.
2. **Information Decay.** The mutual information $I(\mathbf{a}_{t+k}^*; \mathbf{o}_t)$ decreases with k, reflecting the diminishing predictive power of $\mathbf{o}_t$. Consequently, the conditional entropy $H(\Delta\mathbf{a}_k^* \mid \mathbf{o}_t)$ must increase.

Remark on Absolute Stability but Increased Learning Difficulty In contrast to Delta representations, Absolute parameterization is statistically robust to horizon scaling: the regression targets $\mathbf{a}_{t+k}$ remain confined to a bounded workspace, preventing variance growth. Moreover, Absolute predictions map observations directly to global task-space coordinates, avoiding the stale-reference effect inherent in relative corrections. However, this robustness comes at the cost of substantially increased learning difficulty. Because the policy must implicitly infer full-scene geometry, global localization, and workspace-scale structure from raw observations, Absolute parameterization demands a more complex perceptual model and is often harder to train effectively than Delta-based variants.

G.3 Research Question on Spatial Reparameterization

Beyond temporal structure, the choice of spatial action manifold also imposes strong inductive biases on both the learnability and stability of the control policy. Whereas temporal parameterization governs how predictions evolve over the horizon, spatial parameterization determines the geometry of the control interface itself and thus fundamentally affects the numerical conditioning and perceptual complexity of the policy. **Task-space** control offers a geometrically meaningful abstraction but introduces the potentially ill-conditioned pseudo-inverse Jacobian $\mathbf{J}^\dagger(\mathbf{q}_t)$ into $\mathcal{T}_{\text{total}}$. In contrast, **Joint-space** control is numerically stable by construction, yet forces the policy to regress visual observations into a highly non-linear configuration space, significantly increasing perceptual complexity.

G.4 Summary

Together, these analyses reveal that action parameterization fundamentally governs a core trade-off between **learnability** (the functional complexity of the mapping $f : \mathbf{o}_t \to \mathbf{Z}_t$) and **stability** (the numerical conditioning of the final control transformation $\mathcal{T}_{\text{total}}$). We structure our empirical study around two orthogonal axes: temporal structure and spatial manifold, aiming to identify action parameterizations that most effectively facilitate robust visuomotor policy learning.

H Detailed Statistics

In this section, we provide all detailed statistics of our reported results.

Table 4: Task success rates under different data regimes

Method	Control	# Data	Cube	Pick Cup	Pick and Place	Bimanual Transfer	Average
DP	abs-ee	100	58.70	60.87	69.57	56.52	61.41
	rel-ee		80.43	84.78	69.57	45.65	70.11
	abs-joint		78.26	60.87	60.87	57.61	64.40
	rel-joint		86.96	89.13	73.91	58.70	77.17
	abs-ee	250	83.70	65.22	73.91	65.22	72.01
	rel-ee		91.30	97.83	84.78	76.09	87.50
	abs-joint		95.65	78.26	81.52	75.00	82.61
	rel-joint		96.74	97.83	93.48	75.00	90.76
	abs-ee	500	93.48	73.91	76.09	78.26	80.43
	rel-ee		86.96	100.00	97.83	71.74	89.13
	abs-joint		100.00	65.22	85.87	85.65	84.18
	rel-joint		100.00	97.83	91.30	84.78	93.48
ACT	abs-ee	100	76.09	54.35	57.61	59.78	61.96
	rel-ee		93.48	80.56	70.65	57.61	75.57
	abs-joint		70.65	75.00	55.43	43.48	61.14
	rel-joint		91.30	77.78	67.39	42.39	69.72
	abs-ee	250	77.17	63.04	66.30	63.04	67.39
	rel-ee		85.87	97.83	84.78	71.70	85.04
	abs-joint		77.17	84.78	70.65	52.17	71.20
	rel-joint		84.78	95.65	83.70	69.57	83.42
	abs-ee	500	96.74	76.09	56.52	68.48	74.46
	rel-ee		97.83	100.00	96.74	65.22	89.95
	abs-joint		98.91	95.65	77.17	72.83	86.14
	rel-joint		100.00	100.00	89.13	76.09	91.30

Table 5: Task success rates under different training epochs

Method	Control	# Epochs	Cube	Pick Cup	Pick and Place	Bimanual Transfer	Average
ACT	abs-ee	300	66.30	52.38	66.30	61.96	61.74
	rel-ee		73.91	92.86	91.30	67.39	81.37
	abs-joint		65.22	64.29	67.39	51.09	62.00
	rel-joint		82.61	95.24	92.39	64.13	83.59
	abs-ee	600	77.17	63.04	66.30	63.04	67.39
	rel-ee		85.87	97.83	84.78	67.39	83.97
	abs-joint		77.17	84.78	70.65	52.17	71.20
	rel-joint		84.78	95.65	83.70	69.57	83.42
	abs-ee	900	80.43	65.22	58.70	71.74	69.02
	rel-ee		85.87	92.39	82.61	66.30	81.79
	abs-joint		83.70	73.91	61.96	58.70	69.57
	rel-joint		82.61	95.65	89.13	68.48	83.97
	abs-ee	1200	83.70	71.74	68.48	72.83	74.18
	rel-ee		86.96	97.83	89.13	71.74	86.41
	abs-joint		86.96	86.96	68.48	66.30	77.17
	rel-joint		84.78	100.00	95.65	67.39	86.96
DP	abs-ee	300	79.35	58.70	65.22	65.22	67.12
	rel-ee		93.48	100.00	82.61	75.00	87.77
	abs-joint		90.22	65.22	67.39	63.04	71.47
	rel-joint		96.74	95.65	91.30	67.39	87.77
	abs-ee	600	83.70	65.22	73.91	65.22	72.01
	rel-ee		91.30	97.83	84.78	76.09	87.50
	abs-joint		95.65	78.26	81.52	75.00	82.61
	rel-joint		96.74	97.83	93.48	75.00	90.76
	abs-ee	900	66.30	67.39	80.43	81.52	73.91
	rel-ee		88.04	89.13	82.61	70.65	82.61
	abs-joint		88.04	73.91	75.00	73.91	77.72
	rel-joint		95.65	97.83	86.96	78.26	89.67
	abs-ee	1200	80.43	69.57	78.26	79.35	76.90
	rel-ee		95.65	95.65	92.39	71.74	88.86
	abs-joint		93.48	76.09	76.09	73.91	79.89
	rel-joint		97.83	100.00	100.00	79.35	94.29

Table 6: Task scores averaged across all single-task settings (data and epoch).

Method	Space	Mode	Cube	Pick Cup	Pick and Place	Bimanual Transfer	Average
ACT	EEF	abs	79.68 ± 2.45	63.44 ± 2.31	62.93 ± 1.79	66.03 ± 2.14	68.02 ± 3.95
		delta	87.18 ± 1.86	95.03 ± 1.79	85.61 ± 2.28	65.84 ± 1.78	83.41 ± 6.21
	Joint	abs	79.76 ± 2.63	79.44 ± 3.17	67.47 ± 1.69	56.42 ± 3.00	70.77 ± 5.57
		delta	87.16 ± 2.11	94.84 ± 1.84	85.74 ± 2.45	65.24 ± 2.78	83.24 ± 6.33
ACT	EEF	abs	79.68 ± 2.45	63.44 ± 2.31	62.93 ± 1.79	66.03 ± 2.14	68.02 ± 3.95
		delta	87.18 ± 1.86	95.03 ± 1.79	85.61 ± 2.28	65.84 ± 1.78	83.41 ± 6.21
	Joint	abs	79.76 ± 2.63	79.44 ± 3.17	67.47 ± 1.69	56.42 ± 3.00	70.77 ± 5.57
		delta	87.16 ± 2.11	94.84 ± 1.84	85.74 ± 2.45	65.24 ± 2.78	83.24 ± 6.33

Table 7: 10-task performance of ACT across different data settings

Task	10 data				25 data				50 data				100 data			
	abs-ee	rel-ee	abs-qpos	rel-qpos	abs-ee	rel-ee	abs-qpos	rel-qpos	abs-ee	rel-ee	abs-qpos	rel-qpos	abs-ee	rel-ee	abs-qpos	rel-qpos
Adjust Bottle	20.00	50.00	20.00	75.00	70.00	50.00	40.00	50.00	70.00	40.00	80.00	90.00	90.00	100.00	80.00	90.00
Dump Bin Bigbin	0.00	40.00	10.00	10.00	10.00	50.00	10.00	40.00	40.00	60.00	50.00	40.00	90.00	60.00	70.00	60.00
Grab Roller	0.00	30.00	0.00	40.00	0.00	50.00	40.00	100.00	60.00	10.00	100.00	100.00	50.00	80.00	90.00	100.00
Lift Pot	10.00	20.00	0.00	0.00	20.00	30.00	50.00	10.00	30.00	0.00	40.00	30.00	40.00	30.00	70.00	30.00
Move Playingcard Away	10.00	0.00	0.00	10.00	20.00	20.00	20.00	50.00	40.00	30.00	30.00	50.00	20.00	40.00	10.00	50.00
Open Laptop	20.00	20.00	30.00	40.00	40.00	40.00	40.00	20.00	20.00	40.00	30.00	40.00	20.00	50.00	60.00	70.00
Place Burger Fries	10.00	10.00	10.00	0.00	0.00	20.00	20.00	10.00	20.00	0.00	30.00	50.00	20.00	50.00	30.00	30.00
Place Container Plate	0.00	0.00	10.00	0.00	0.00	0.00	0.00	30.00	20.00	40.00	10.00	30.00	40.00	80.00	40.00	60.00
Press Stapler	0.00	20.00	0.00	10.00	0.00	50.00	20.00	0.00	10.00	30.00	0.00	40.00	40.00	30.00	20.00	50.00
Shake Bottle Horizontally	60.00	40.00	50.00	50.00	100.00	60.00	70.00	50.00	70.00	70.00	100.00	70.00	80.00	90.00	60.00	100.00
Average	13.00	23.00	13.00	24.00	26.00	37.00	31.00	36.00	38.00	32.00	47.00	54.00	49.00	61.00	53.00	64.00

Table 8: 10-task performance of DP across different data settings

Task	10 data				25 data				50 data				100 data			
	abs-ee	rel-ee	abs-qpos	rel-qpos	abs-ee	rel-ee	abs-qpos	rel-qpos	abs-ee	rel-ee	abs-qpos	rel-qpos	abs-ee	rel-ee	abs-qpos	rel-qpos
Adjust Bottle	40.00	50.00	30.00	30.00	80.00	80.00	50.00	60.00	40.00	90.00	60.00	80.00	80.00	100.00	70.00	100.00
Dump Bin Bigbin	10.00	30.00	10.00	10.00	0.00	40.00	20.00	50.00	10.00	60.00	50.00	50.00	70.00	40.00	70.00	50.00
Grab Roller	0.00	30.00	60.00	20.00	20.00	100.00	40.00	70.00	10.00	70.00	80.00	90.00	60.00	80.00	80.00	100.00
Lift Pot	20.00	20.00	10.00	0.00	0.00	10.00	20.00	10.00	10.00	10.00	50.00	60.00	20.00	40.00	50.00	70.00
Move Playingcard Away	10.00	0.00	0.00	0.00	10.00	20.00	30.00	40.00	20.00	30.00	40.00	40.00	30.00	40.00	10.00	80.00
Open Laptop	20.00	20.00	70.00	20.00	30.00	20.00	30.00	40.00	20.00	60.00	60.00	70.00	30.00	20.00	60.00	80.00
Place Burger Fries	0.00	0.00	0.00	0.00	10.00	0.00	40.00	10.00	10.00	30.00	70.00	30.00	30.00	30.00	40.00	50.00
Place Container Plate	0.00	10.00	10.00	0.00	0.00	0.00	10.00	30.00	40.00	60.00	40.00	50.00	20.00	30.00	70.00	50.00
Press Stapler	0.00	10.00	0.00	20.00	10.00	10.00	20.00	30.00	20.00	50.00	30.00	40.00	20.00	50.00	40.00	50.00
Shake Bottle Horizontally	50.00	50.00	50.00	30.00	100.00	60.00	80.00	30.00	100.00	90.00	80.00	70.00	100.00	90.00	80.00	80.00
Average	15.00	22.00	24.00	13.00	26.00	34.00	34.00	37.00	28.00	55.00	56.00	58.00	46.00	52.00	57.00	71.00

Table 9: Task performance of ACT across different epoch settings

Task	25w				50w				75w				100w			
	abs-ee	rel-ee	abs-qpos	rel-qpos	abs-ee	rel-ee	abs-qpos	rel-qpos	abs-ee	rel-ee	abs-qpos	rel-qpos	abs-ee	rel-ee	abs-qpos	rel-qpos
Adjust Bottle	0.30	0.40	0.20	0.40	0.40	0.40	0.20	0.40	0.40	0.20	0.30	0.40	0.70	0.30	0.70	0.40
Dump Bin Bigbin	0.00	0.30	0.60	0.50	0.40	0.90	0.50	0.50	0.40	0.30	0.40	0.30	0.40	0.30	0.60	0.60
Grab Roller	0.20	0.20	0.80	1.00	0.50	0.50	0.30	0.90	0.00	0.50	0.60	0.90	0.60	0.70	0.60	0.90
Lift Pot	0.00	0.00	0.40	0.10	0.00	0.00	0.20	0.40	0.60	0.10	0.40	0.20	0.30	0.00	0.20	0.30
Move Playingcard Away	0.00	0.00	0.00	0.40	0.10	0.10	0.20	0.40	0.00	0.40	0.50	0.30	0.10	0.50	0.10	0.30
Open Laptop	0.10	0.20	0.30	0.10	0.10	0.50	0.40	0.40	0.10	0.20	0.20	0.20	0.20	0.50	0.20	0.30
Place Burger Fries	0.10	0.00	0.20	0.40	0.20	0.30	0.40	0.40	0.10	0.30	0.30	0.60	0.00	0.30	0.10	0.50
Place Container Plate	0.30	0.20	0.60	0.70	0.20	0.50	0.60	0.50	0.30	0.50	0.40	0.60	0.20	0.60	0.30	0.50
Press Stapler	0.20	0.10	0.40	0.10	0.10	0.20	0.20	0.40	0.10	0.10	0.10	0.40	0.20	0.10	0.40	0.30
Shake Bottle Horizontally	0.80	0.80	0.60	0.70	1.00	0.90	1.00	0.70	1.00	0.90	0.70	0.60	1.00	0.80	1.00	0.50
Average	0.20	0.22	0.41	0.44	0.30	0.43	0.40	0.50	0.30	0.35	0.39	0.45	0.37	0.41	0.42	0.46

Table 10: Task performance of DP across different epoch settings

Task	25w				50w				75w				100w			
	abs-ee	rel-ee	abs-qpos	rel-qpos	abs-ee	rel-ee	abs-qpos	rel-qpos	abs-ee	rel-ee	abs-qpos	rel-qpos	abs-ee	rel-ee	abs-qpos	rel-qpos
Adjust Bottle	0.60	0.20	0.30	0.70	0.40	0.40	0.10	0.60	0.20	0.30	0.20	0.50	0.40	0.40	0.50	0.90
Dump Bin Bigbin	0.10	0.50	0.30	0.60	0.10	0.70	0.70	0.40	0.30	0.40	0.70	0.60	0.50	0.50	0.80	0.50
Grab Roller	0.60	0.30	0.50	0.70	0.10	0.90	0.10	0.90	0.20	0.80	0.40	0.80	0.60	0.70	0.50	0.90
Lift Pot	0.00	0.00	0.20	0.00	0.10	0.10	0.10	0.00	0.00	0.30	0.10	0.40	0.60	0.00	0.30	0.10
Move Playingcard Away	0.00	0.20	0.00	0.40	0.20	0.30	0.20	0.70	0.10	0.30	0.30	0.20	0.10	0.20	0.20	0.60
Open Laptop	0.10	0.00	0.00	0.10	0.20	0.30	0.20	0.40	0.10	0.20	0.20	0.20	0.10	0.50	0.40	0.20
Place Burger Fries	0.10	0.10	0.20	0.10	0.10	0.30	0.20	0.40	0.20	0.10	0.30	0.40	0.30	0.20	0.40	0.60
Place Container Plate	0.20	0.20	0.60	0.40	0.40	0.40	0.40	0.70	0.20	0.30	0.50	0.40	0.00	0.50	0.40	0.70
Press Stapler	0.10	0.20	0.10	0.10	0.20	0.20	0.20	0.40	0.20	0.30	0.10	0.40	0.50	0.20	0.40	0.60
Shake Bottle Horizontally	0.80	0.80	0.60	0.90	1.00	1.00	1.00	1.00	0.90	1.00	0.90	1.00	1.00	0.80	1.00	1.00
Average	0.26	0.25	0.28	0.40	0.28	0.46	0.32	0.55	0.24	0.40	0.37	0.49	0.41	0.40	0.49	0.61