Unrolling the Action Manifold: Visuomotor Policy Learning via Recursive Cascades

Jiahang Cao[1,6*] **Zhuoyang Chen**[2,3*] **Kun Wu**[4] **Yulin Li**[5] **Hongliang Lu**[7]
Jiaxu Wang[8] **Jingkai Sun**[1] **Qiang Zhang**[4,9] **Jun Ma**[7] **Qihao Zheng**[6]
Chunfeng Song[6] **Ping Luo**[1] **Andrew F. Luo**[1†]
[1]HKU [2]UMich [3]SJTU [4]X-Humanoid [5]NUS
[6]Shanghai AI Lab [7]HKUST [8]CUHK [9]USTC

https://sagecao1125.github.io/RCP-Site

Abstract: Current generative visuomotor policies often face a trade-off between inference speed and execution stability. Autoregressive architectures are constrained by sequential decoding latency and compounding errors, whereas diffusion models demand computationally expensive iterative denoising. We introduce the Recursive Cascade Policy (RCP), an architecture inspired by human hierarchical cognitive control that frames action generation as recursive temporal infilling directly within explicit physical action space. A single weight-shared Transformer first predicts sparse boundary anchors to sketch a global trajectory, then recursively populates the temporal gaps. By conditioning each step on its immediate geometric anchors and a hierarchically propagated latent, RCP renders intermediate trajectory segments conditionally independent. This structural prior restricts the attention mechanism to a constant-length context, decoupling attention complexity from the total action horizon and unlocking highly efficient parallel decoding. With only 19M parameters, less than one-quarter the size of ACT and Diffusion Policy, RCP achieves superior performance across diverse simulation and real-world tasks. By grounding generation in local geometric boundaries rather than unbounded histories, RCP maintains robust temporal coherence, demonstrating recursive cascades as an efficient and scalable approach for robotic policies.

Keywords: Recursive Generation, Visuomotor Policy, Robot Manipulation

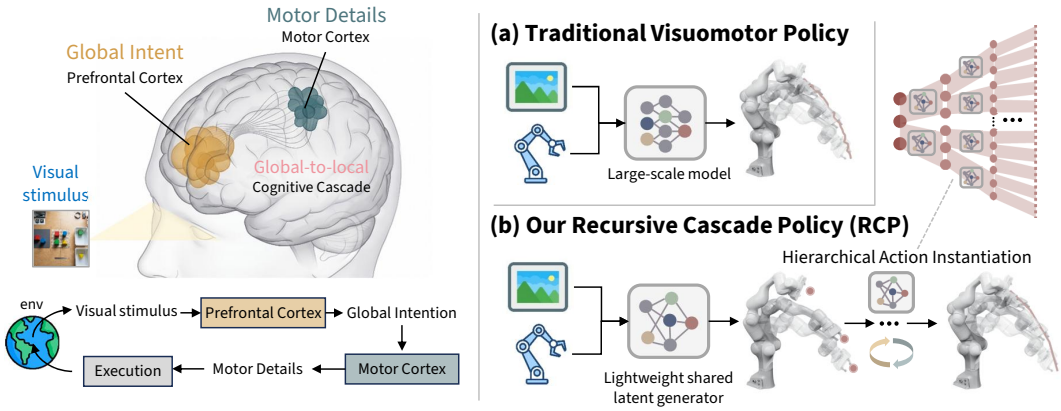

Figure 1: **Motivation and Conceptual Framework.** (Left) Analogous to diverse complex control systems, cognitive control manages task complexity via a cascade that refines high-level intentions into granular motor details. (Right-a) Traditional visuomotor policies employ monolithic, large-scale models to establish a flat, direct observation-to-action mapping. (Right-b) Our Recursive Cascade Policy (RCP) employs a lightweight shared generator to iteratively instantiate a hierarchical action tree. This recursive cascade elegantly resolves spatial uncertainty, delivering high-fidelity control with exceptional architectural economy.

1 Introduction

Scaling robot learning to complex, long-horizon tasks requires generative policies capable of synthesizing fine-grained, high-dimensional action sequences from raw sensory observations [1, 2, 3]. However, existing generative paradigms often require a compromise between inference speed and execution stability. Autoregressive (AR) architectures are constrained by sequential decoding latency and remain highly susceptible to compounding errors over extended execution horizons [4, 5]. Conversely, diffusion models offer excellent long-horizon stability and multimodal expressivity but require computationally expensive iterative denoising, introducing significant latency that can stall real-time control [6].

Hierarchical and coarse-to-fine structures offer a compelling mechanism to resolve this bottleneck through temporal abstraction [7, 8]. Decoupling abstract planning from execution provides a robust safeguard against error accumulation. Analogous principles naturally emerge across diverse complex systems, from the layered control loops of early robotic architectures [9] to the hierarchical cognitive control networks of the human brain [10, 11, 12, 13]. By establishing global trajectory sketches before populating local details, hierarchical policies maintain macro-level stability while isolating micro-motor noise. Yet, current multi-scale deep learning approaches frequently rely on disjointed networks for planning and execution, or operate within ungrounded, discrete latent spaces susceptible to lossy tokenization. Crucially, these models typically scale their attention complexity with the action horizon, failing to fully escape the latency bottleneck of long-sequence generation.

To address these limitations, we introduce the **Recursive Cascade Policy (RCP)**. RCP departs from unidirectional sequence prediction and continuous denoising by framing action generation as recursive temporal infilling directly within the explicit physical action space. Operating through a single weight-shared Transformer, RCP initializes generation by predicting a sparse set of boundary anchors to establish a global sketch of the movement. It then recursively populates the temporal gaps between these anchors to instantiate the complete trajectory. Because each infilling step conditions solely on its immediate geometric boundaries, intermediate trajectory segments become conditionally independent. This structural prior restricts the attention mechanism to a constant-length local context, effectively decoupling attention complexity from the total action horizon. Furthermore, this conditional independence unlocks *parallel trajectory sampling*, allowing the simultaneous generation of entire hierarchical levels via batched forward passes.

We demonstrate the efficacy of the Recursive Cascade Policy across an extensive suite of evaluations, encompassing the diverse manipulation tasks of ManiSkill [14], complex bimanual coordination in RoboTwin [15], and challenging real-world deployments. Despite achieving superior execution fidelity, RCP remains remarkably lightweight. Operating with a mere **19M** parameters, it represents a fourfold reduction compared to ACT [2] and a fivefold reduction relative to Diffusion Policy [1], all while sustaining substantially higher inference throughput. As a generalized formulation, the RCP paradigm natively accommodates varying visual modalities and exhibits predictable capability gains as model capacity scales. Crucially, our method preserves strict temporal coherence and robust performance even as the predicted action chunk horizon extends. To elucidate the internal mechanisms driving this stability, UMAP visualizations of the latent refinement flow provide empirical evidence of how the policy progressively grounds high-level intentions into precise physical executions. Collectively, these results establish recursive cascading as a highly effective foundation for scalable, interpretable, and parameter-efficient robot learning.

2 Related Work

2.1 End-to-End Visuomotor Policy Learning

Visuomotor control has evolved from simple Behavioral Cloning (BC) [16, 17] to massive Vision-Language-Action (VLA) and World-Action Models (WAM), exemplified by RT-1 [18], OpenVLA [19], and Fast-WAM [20]. These systems leverage large-scale pretraining to map multimodal tokens directly into executable actions. While these foundational architectures excel at semantic

generalization, learning both abstract reasoning and high-frequency motor execution within a single unstratified mapping poses significant representational challenges. To improve local precision, generative policies such as Action Chunking Transformers (ACT) [2], Diffusion Policy [1], and the π series [3, 21, 22] predict entire blocks of actions over receding horizons. However, generating dense, high-frequency action chunks in a single "flat" pass creates computational bottlenecks and degrades temporal coherence as prediction horizons expand. Our work retains the execution advantages of action chunking but reframes the internal generation process as a recursive refinement.

2.2 Hierarchical and Multi-scale Robot Control

Hierarchical abstraction is foundational to robotics, evolving from early reactive [9] and hybrid deliberative-reactive architectures [23, 24] to modern multi-scale sequence generation [25]. Today, keyframe-driven architectures condition local trajectories on sparse critical poses [7, 8, 26, 27, 28], while hierarchical generative models utilize structured spatial anchors to maintain temporal coherence [29, 30, 31]. Concurrently, dual-system VLAs [21, 32, 33] explicitly separate abstract planning from low-level controllers. While effective, integrating these distinct modules often requires careful engineering to maintain representational consistency. Recently, CARP [34] advanced this paradigm via a unified coarse-to-fine autoregressive policy that compresses actions using a multi-stage VQ-VAE. While adept at capturing multi-scale dynamics, operating in a discrete space risks quantization artifacts, and generating finer scales requires an expanding attention context that encompasses preceding coarser scales. RCP offers a complementary continuous-space perspective: by restricting attention to a constant-length local context bounded by immediate anchors, RCP circumvents quantization steps and maintains a bounded computational footprint at every cascade level.

2.3 Human-inspired Cognitive Control and Cascade Models

Since Lashley's critique of linear behavioral chaining [35], cognitive science has established that complex motor sequences are governed by hierarchical plans. In human motor control, this complexity is managed via an organized cascade within the prefrontal cortex [36, 10]. Computational models of cognition demonstrate that structural decoupling, where higher-order regions disambiguate temporally distant goals before lower-level regions select local motor commands [37, 12]. Recent work spanning both biological and artificial agents highlight that multi-scale temporal abstractions are paramount for robust motor control [13]. While many robotic systems mimic the functional separation of these planning levels, RCP explicitly formalizes their fluid, recursive refinement. By employing a weight-shared generator, RCP iteratively populates the action trajectory, progressively instantiating dense motor programs from a sparse global sketch.

3 Methodology

The core philosophy of Recursive Cascade Policy (RCP) is to transform the traditional monolithic action prediction task into a structured process of uncertainty reduction. By mimicking the human hierarchical control to refine abstract intentions into motor programs, RCP employs a weight-shared architecture to iteratively instantiate action trajectories across a gradient of abstraction.

3.1 System Overview

The RCP framework consists of two primary components: an Observation Encoder E_ϕ and a Recursive Shared Generator G_θ (Fig. 2(a)). Unlike flat architectures that resolve all temporal scales simultaneously, RCP operates through a sequence of refinement levels $L \in \{0, 1, \ldots, N\}$. At each level, a weight-shared generator (6M) produces a set of action latents that are increasingly grounded in physical detail. This hierarchical progression (Fig. 2(b)) ensures representational consistency and mitigates compounding errors, as the global intent established at Level 0 serves as a stable anchor for the local motor adjustments generated in subsequent passes.

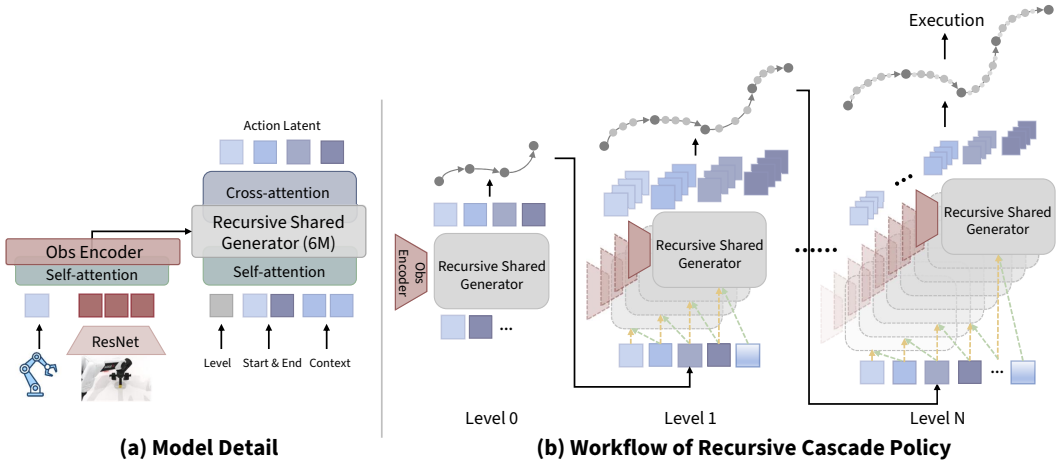

Figure 2: **Architecture and Recursive Workflow.** (a) The Observation Encoder processes visual and proprioceptive inputs into unified state embeddings via a self-attention mechanism. These embeddings serve as queries for the weight-shared Recursive Shared Generator, which integrates environmental context with action latents through cross-attention. (b) Action generation is structured as a hierarchical evolution across increasing temporal resolutions. Level 0 synthesizes sparse trajectory anchors to establish global task intent. These anchors are then recursively refined through the shared generator to produce high-frequency motor commands.

3.2 Observation Encoding

To provide a rich contextual foundation for action generation, the Observation Encoder E_ϕ maps multi-modal inputs into a unified token sequence. The image observation I_t is processed through a ResNet backbone and flattened into a sequence of spatial tokens $\mathbf{t}_{\texttt{spatial}}$ enriched with 2D sinusoidal positional encodings. Concurrently, the robot proprioceptive state p_t is projected into a distinct token $\mathbf{t}_{\texttt{proprio}}$. These heterogeneous tokens are concatenated and contextualized through self-attention layers to form the comprehensive observation sequence: $\mathbf{t}_{\texttt{obs}} = \texttt{SelfAttention}\left([\mathbf{t}_{\texttt{proprio}}, \mathbf{t}_{\texttt{spatial}}]\right)$. $\mathbf{t}_{\texttt{obs}}$ acts as the external key-value foundation for the subsequent cross-attention mechanisms within the generator.

3.3 The Recursive Shared Generator

Architecture of Recursive Shared Generator. The core computational engine of our architecture is the Recursive Shared Generator G_θ, which operates as a pure embedding-to-embedding Transformer designed to infill intermediate trajectory states. For any given segment at refinement level l, the generator first assembles a unified input sequence $\mathbf{x}^l$:

$$\mathbf{x}^l = [\mathbf{t}^l_{\texttt{level}}, \mathbf{t}^{l-1}_{\texttt{latent}}, \mathbf{t}^l_{\texttt{start}}, \mathbf{t}^l_{\texttt{end}}, \mathbf{t}^l_{\texttt{mask}}]. \tag{1}$$

Within this sequence, a level embedding $\mathbf{t}_{\texttt{level}}$ to indicate the current abstraction scale, and a start token $\mathbf{t}_{\texttt{start}}$ that is encoded from the current robot state. To empower the model with open-ended extrapolation capabilities, the terminal boundary $\mathbf{t}_{\texttt{end}}$ is stochastically fused with an open-end token via a structural classifier-free guidance [38] strategy during training. $\mathbf{t}_{\texttt{mask}}$ comprises $k-1$ placeholders for intermediate waypoints, where k denotes the number of temporal sub-intervals generated per recursive step. The token $\mathbf{t}_{\texttt{latent}}$ carries an inherited semantic prior from the parent level (detailed in Eq. 4), which is explicitly initialized as a null token at the root generation level ($l = 0$).

After injecting standard positional encodings, the sequence x^l propagates through N identical Transformer blocks. Each block executes a streamlined computational flow:

$$\mathbf{x}^l = \texttt{SelfAttention}(\mathbf{x}^l), \tag{2}$$

$$\mathbf{z}^l = \texttt{FFN}\big(\texttt{CrossAttention}(\texttt{Query} = \mathbf{x}^l, \texttt{Key/Value} = \mathbf{t}_{\texttt{obs}})\big), \tag{3}$$

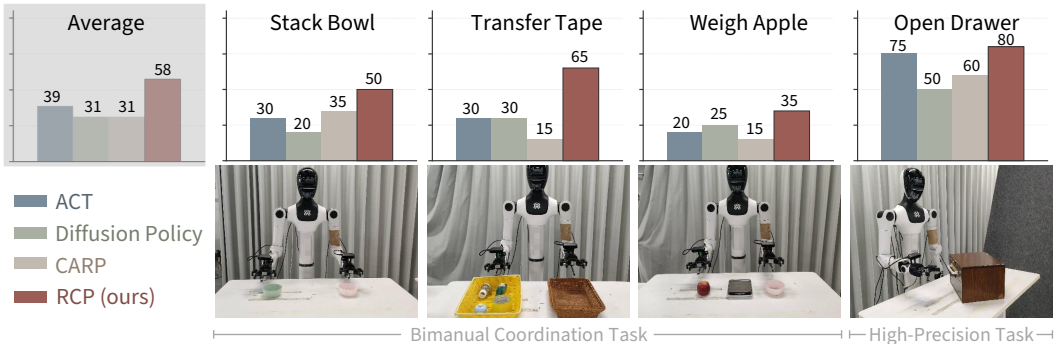

Figure 3: **Quantitative Efficacy in Real-World Deployments.** Success rates evaluated on the dual-arm humanoid platform. Our RCP architecture systematically dominates both monolithic (ACT, DP) and discrete multi-scale (CARP) baselines, achieving a superior 58% average success rate and exhibiting pronounced robustness across diverse physical manipulation paradigms.

where first a multi-head self-attention layer establishes internal temporal consistency among the trajectory tokens; a cross-attention layer then grounds these tokens into the physical environment by attending over the external observation sequence $\mathbf{t}_{\texttt{obs}}$; and a standard Feed-Forward Network (FFN) concludes the block. Following the final layer, the generator directly extracts the updated representations corresponding to the start and mask positions to formulate the whole waypoint latent.

Latent Aggregation Module. To guarantee semantic continuity for subsequent recursive subdivisions, a Latent Aggregation module synthesizes the global profile of the current segment. This module employs a learnable query $\mathbf{q}$ to dynamically attend over the generated waypoint latent $\mathbf{z}^l$:

$$\mathbf{t}^l_{\texttt{latent}} = \texttt{CrossAttention}(\texttt{Query} = \mathbf{q}, \texttt{Key/Value} = \mathbf{z}^l). \tag{4}$$

The resulting aggregated vector $\mathbf{t}^l_{\texttt{latent}}$ encapsulates the holistic abstract execution profile of the current trajectory span, serving as the robust parent prior to gracefully bias and guide the child segments in the next refinement level.

3.4 Hierarchical Action Instantiation

The workflow of RCP is defined by the recursive application of the shared generator G_θ to populate the action manifold, which is illustrated in Fig. 2(b).

Level 0: The Gist of Intent. In the initial pass, G_θ receives the global boundaries of the action chunk. It produces a sparse sequence of K action latents representing the global trajectory sketch. These latents are passed through an action decoder D_ψ to yield the coarse action sequence:

$$A^0 = D_\psi(\mathbf{z}^0) = \{a^0_1, a^0_2, \ldots, a^0_K\}. \tag{5}$$

This level establishes the macro-task logic, providing a stable backbone for the entire movement.

Recursive Refinement: Learning to Grasp. For levels $l > 0$, the previously generated actions serve as relational anchors. Each adjacent pair of actions $(a^{l-1}_i, a^{l-1}_{i+1})$ is recursively re-encoded and fed back into G_θ as the new $\mathbf{t}_{\texttt{start}}$ and $\mathbf{t}_{\texttt{end}}$ tokens. The generator then populates the gaps between these anchors with higher-frequency action latents. This process repeats until Level N, where the model outputs the final fine-grained trajectory $\mathcal{A}^{\texttt{fine}}$ (*i.e.*, $\mathcal{A}^N$) ready for execution. By reusing the same generator across levels, RCP maintains a unified representational space, allowing the model to refine the global strategy based on local feasibility.

3.5 Training and Inference

Unified Cascaded Training. We optimize the shared recursive generator G_θ through a unified training paradigm with two modes that balance local execution precision with hierarchical stability.

Table 1: **Quantitative Results on the RoboTwin 2.0.** We report the success rates across 10 challenging bimanual manipulation tasks. The **1st** and **2nd** results are highlighted.

Method	beat_block_hammer	hanging_mug	place_dual_shoes	place_phone_stand	place_bread_basket	put_object_cabinet	place_fan	place_cont._plate	move_can_pot	pick_div._bottles	Average
Diffusion Policy [1]	0.42	0.08	0.08	0.13	0.14	0.42	0.03	0.41	0.39	0.05	0.22
ACT [2]	0.56	0.07	0.09	0.02	0.06	0.15	0.01	0.72	0.22	0.07	0.20
π_0 [3]	0.43	0.11	0.15	0.35	0.17	**0.68**	0.20	**0.88**	**0.58**	**0.27**	0.38
RDT [39]	0.77	0.23	0.04	0.15	0.10	0.33	0.12	0.78	0.25	0.02	0.28
RCP (ours)	**0.81**	**0.31**	**0.20**	**0.42**	**0.33**	0.40	**0.26**	0.81	0.54	0.24	**0.43**

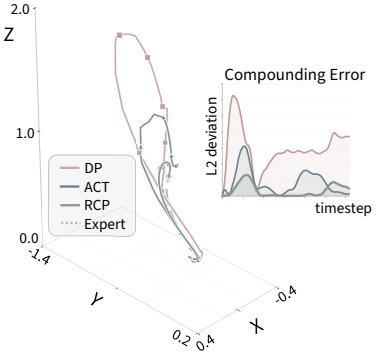

Figure 4: **Trajectory Analysis.** Our RCP tightly tracks the expert (left), effectively bounding temporal L2 deviation and mitigating compounding errors (right).

Table 2: **Quantitative Results on ManiSkill 3.** RCP demonstrates superior performance (0.93) with 100 demonstrations, outperforming baselines (*e.g.*, VAs and large-scale VLAs) trained on 1k demonstrations.

Method	Demos	PickCube-v1	PushCube-v1	StackCube-v1	Average
BC	100	0.00	0.00	0.00	0.00
	1000	0.03	0.81	0.00	0.28
ACT [2]	100	0.28	0.30	0.33	0.30
	1000	0.98	0.89	0.80	0.89
Diffusion Policy [1]	100	0.76	0.41	0.61	0.59
	1000	1.00	0.86	0.81	0.89
DP + VGGT [40]	1000	0.96	0.91	0.65	0.84
PAR [41]	1000	0.73	1.00	0.48	0.74
OpenVLA [19]	1000	0.08	0.08	0.08	0.08
Octo [42]	1000	0.00	0.00	0.00	0.00
RDT [39]	1000	0.77	1.00	0.74	0.84
RCP (ours)	100	**0.79**	**1.00**	**0.99**	**0.93**

(a) The *segment mode* uniformly samples a level l and a segment s, conditioning the generator strictly on ground-truth boundaries to predict intermediate waypoints $\hat{\mathcal{A}}_s^l$. Regulated by $\mathcal{L}_{\texttt{segment}} = \|\mathcal{A}_s^l - \hat{\mathcal{A}}_s^l\|_1$, this teacher-forced supervision guarantees high-fidelity local execution (see App. 1).

(b) Conversely, the *cascade mode* unrolls the entire action tree from root to leaves, dynamically conditioning intermediate levels on the *predicted* waypoints of their parent nodes. Evaluated as $\mathcal{L}_{\texttt{cascade}} = \sum_{l=0}^{L-1} \lambda_l \sum_{s \in l} \|\mathcal{A}_s^l - \hat{\mathcal{A}}_s^l\|_1$, where λ_l is the weight of level l's contribution. This mode explicitly mitigates the compounding errors propagated down the hierarchy (see App. 2).

To synergize these complementary mechanisms, during training, these two modes alternate across epochs with a frequency-division factor α to optimize both objectives.

Parallel Trajectory Sampling. A profound architectural advantage of RCP emerges during real-time robotic execution. Traditional autoregressive models require sequential decoding steps that scale linearly with the total trajectory length. Conversely, RCP naturally supports highly efficient parallel sampling. At any given refinement depth l, the generation of the intermediate sub-segments is conditionally independent given their respective boundary anchors. Consequently, all segments occupying the same hierarchical depth can be batched and evaluated simultaneously in a single network forward pass. This reduces the requisite network evaluations from an exponential sequence to an $\mathcal{O}(L)$ temporal complexity, enabling RCP to generate exceptionally long-horizon and high-frequency control signals within strict latency constraints. The algorithm is illustrated in App. 3.

4 Experiments

Our experimental evaluation aims to address three fundamental questions: (1) Does the recursive cascade paradigm enhance performance across diverse manipulation tasks compared to monolithic policies (*e.g.*, ACT, Diffusion Policy)? (2) How does RCP maintain robustness and efficiency as the action chunk horizon scales? (3) Does the internal latent space of the shared generator exhibit a hierarchical structure that aligns with our cognitive motivation?

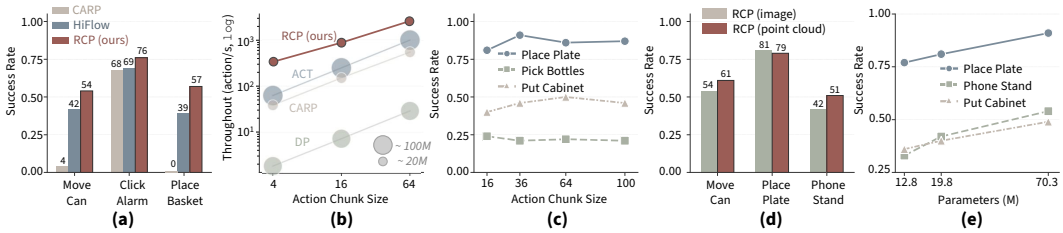

Figure 5: **Comprehensive evaluation of our RCP.** (a) Success rate comparison against the multi-scale baselines across distinct manipulation tasks. (b) Inference throughput ($\log_{10}$) scaling with increasing action chunk sizes, highlighting RCP's efficiency advantage. (c) Ablation on action chunk scaling, showing RCP's robustness. (d) Generalization of the RCP paradigm across different visual input modalities. (e) Performance scaling behavior across small, base, and large parameter regimes.

4.1 Comparative Evaluation

RoboTwin. We evaluate RCP on the long-horizon bimanual tasks of RoboTwin 2.0 [15]. As shown in Tab. 1, RCP achieves a 43.2% average success rate, outperforming monolithic baselines like Diffusion Policy (21.5%) and ACT (19.7%) by a large margin, and even surpassing the massive VLA π_0 (38.2%) at a fraction of the computational cost. Fig. 4 confirms that this superior performance stems from enhanced execution stability, as RCP tightly tracks expert trajectories and bounds temporal L2 deviation to successfully eliminate compounding errors.

ManiSkill. We evaluate RCP on precision-critical tasks within ManiSkill 3. As shown in Tab. 2, our compact policy achieves a 93.0% average success rate, outperforming ACT (89.0%), DP (89.0%), and RDT (84.0%) even when these baselines are trained on 10× more demonstrations. These results confirm that recursive latent progression provides a superior paradigm, mastering complex trajectories without requiring massive parameter counts or excessive data.

Real-world Tasks. Evaluated on the dual-arm Tien Kung 2.0 humanoid platform (Fig. 3), RCP achieves a commanding 58% average success rate, outperforming baselines by 19%. Specifically, RCP excels in both bimanual coordination (65% on `Transfer Tape`) and high-precision articulation (80% on `Open Drawer`), conclusively validating its robust physical execution.

4.2 Ablation Studies

Policy Execution Superiority. We evaluate RCP against CARP [34] and HiFlow [43], two representative multi-scale action baselines. As illustrated in Fig. 5(a), our method delivers vastly superior execution reliability across diverse manipulation tasks. In the highly demanding Move Can scenario, RCP achieves a 54% success rate compared to a 42% for HiFlow and a mere 8% for CARP.

Inference Efficiency. A primary advantage of RCP is its computational efficiency during execution. Fig. 5(b) benchmarks throughput against established methods across increasing action chunk sizes. While dense iterative models suffer from severe latency degradation, RCP maintains exceptionally high throughput (*e.g.*, 55 Hz for the 16-action chunk) due to its linear complexity.

Horizon Scaling Robustness. We investigate the stability of RCP across extending prediction horizons by varying the action chunk size from 16 to 100. As illustrated in Fig. 5(c), execution fidelity remains remarkably consistent across all evaluated tasks regardless of the generated sequence length, confirming the inherent temporal robustness of the RCP framework.

Modality Independence. We investigate the architectural flexibility of RCP regarding distinct visual representations. Because RCP functions as a general policy paradigm rather than a modality-specific feature extractor, it natively accommodates diverse sensory inputs. Fig. 5(d) reports the competitive success rates when conditioning the policy on 2D images versus 3D point clouds.

Capacity Scaling. We characterize the scaling properties of RCP by training three model variants: small (12.8M), base (19.8M), and large (70.3M). Fig. 5(e) maps the success rates on three evaluation tasks as a function of parameter count. The consistent upward trajectory across all tasks verifies predictable scaling behavior, positioning the recursive cascade architecture as a highly promising paradigm for deployment within next-generation, large-scale robot foundation models.

4.3 Qualitative Analysis and Interpretability

Visualizing Hierarchical Spatial Refinement. To empirically validate the generation mechanism, we visualize the unrolling of 2D trajectories (Fig. 6) across recursive cascades. At the initial level, the model predicts a sparse set of global anchors that delineate the macroscopic intent. As the cascade descends to the local level, these structural anchors are systematically populated with dense, granular motor actions. This explicit spatial dependency provides clear evidence of the hierarchical refinement process intrinsic to RCP, demonstrating how broad trajectory priors are grounded into precise physical executions.

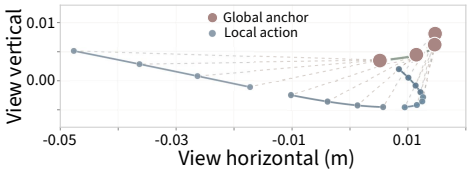

Figure 6: **Visualization of hierarchical action instantiation.** Global anchors establish the macroscopic intent, guiding the dense sequence of local actions.

Latent Representation Analysis. To elucidate the internal mechanisms of our hierarchical cascade, we visualize its latent action space using UMAP. Fig. 7 delineates the spatial relationship between the initial global waypoint latents, represented as distinct nodes, and the continuous distribution of their recursively generated intermediate sub-segments, depicted as the shaded density manifold. The projection exposes a highly structured representational topology. The sparse global anchors first establish a macroscopic trajectory sketch. Driven by the shared generator, these global latents systematically transition along a clear refinement flow, ultimately converging into the dense local execution manifolds highlighted in the inset. This observable dynamic provides direct mechanistic evidence of how the architecture progressively grounds high-level spatial intentions into high-resolution physical executions.

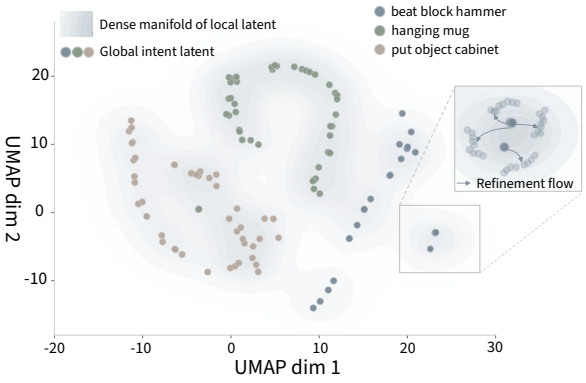

Figure 7: **UMAP projection of the latent action space.** Initial predictions establish a global intent, which are systematically guided along a clear refinement direction into precise local motor actions.

5 Limitations

The current RCP has several limitations. First, our framework currently utilizes fixed temporal intervals for action anchoring. Future research should investigate adaptive cascade structures that learn to identify semantic key latent or goal representation. Second, although RCP exhibits exceptional performance in a compact parameter regime, its integration with the latest VLA and WAM models has yet to be explored. Future work will investigate RCP as a unified pre-training or fine-tuning protocol to bridge the gap between massive semantic priors and high-fidelity robotic execution.

6 Conclusion

In this work, we introduced the Recursive Cascade Policy as a fundamental architectural shift for visuomotor policy learning. By formulating trajectory synthesis through structured hierarchical cascades rather than sequential autoregression or iterative denoising, our method resolves the enduring conflict between inference speed and execution fidelity. We demonstrated that framing action generation as conditionally independent temporal infilling allows a highly compact model to achieve efficient parallel decoding. Extensive experiments on both simulation and real-world demonstrate RCP's robustness across extended prediction horizons and diverse modalities. As the field transitions toward massive foundation models, the recursive paradigm offers a proven and computationally viable pathway to deploy real-time embodied intelligence in complex physical environments.

Acknowledgments

If a paper is accepted, the final camera-ready version will (and probably should) include acknowledgments. All acknowledgments go at the end of the paper, including thanks to reviewers who gave useful comments, to colleagues who contributed to the ideas, and to funding agencies and corporate sponsors that provided financial support.

References

[1] 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 (IJRR)*, 44(10-11):1684–1704, 2025.

[2] 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.

[3] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. pi0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[4] 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)*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[5] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson. Implicit behavioral cloning. In *Conference on Robot Learning (CoRL)*, pages 158–168. PMLR, 2022.

[6] A. Prasad, K. Lin, J. Wu, L. Zhou, and J. Bohg. Consistency policy: Accelerated visuomotor policies via consistency distillation. *arXiv preprint arXiv:2405.07503*, 2024.

[7] E. Johns. Coarse-to-fine imitation learning: Robot manipulation from a single demonstration. In *International Conference on Robotics and Automation (ICRA)*, pages 4613–4619. IEEE, 2021.

[8] S. James, K. Wada, T. Laidlow, and A. J. Davison. Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13739–13748, 2022.

[9] R. Brooks. A robust layered control system for a mobile robot. *IEEE Journal on Robotics and Automation*, 2(1):14–23, 2003.

[10] E. Koechlin, C. Ody, and F. Kouneiher. The architecture of cognitive control in the human prefrontal cortex. *Science*, 302(5648):1181–1185, 2003.

[11] D. Badre. Cognitive control, hierarchy, and the rostro–caudal organization of the frontal lobes. *Trends in Cognitive Sciences*, 12(5):193–200, 2008.

[12] D. Badre, J. Hoffman, J. W. Cooney, and M. D'esposito. Hierarchical cognitive control deficits following damage to the human frontal lobe. *Nature Neuroscience*, 12(4):515–522, 2009.

[13] J. Merel, M. Botvinick, and G. Wayne. Hierarchical motor control in mammals and machines. *Nature Communications*, 10(1):5489, 2019.

[14] S. Tao, F. Xiang, A. Shukla, Y. Qin, X. Hinrichsen, X. Yuan, C. Bao, X. Lin, Y. Liu, T.-k. Chan, et al. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *arXiv preprint arXiv:2410.00425*, 2024.

[15] T. Chen, Z. Chen, B. Chen, Z. Cai, Y. Liu, Z. Li, Q. Liang, X. Lin, Y. Ge, Z. 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.

[16] M. Bain and C. Sammut. A framework for behavioural cloning. In *Machine Intelligence*, pages 103–129, 1995.

[17] N. M. Shafiullah, Z. Cui, A. A. Altanzaya, and L. Pinto. Behavior transformers: Cloning k modes with one stone. *Advances in Neural Information Processing Systems (NeurIPS)*, 35: 22955–22968, 2022.

[18] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *Robotics: Science and Systems (RSS)*, 2023.

[19] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[20] T. Yuan, Z. Dong, Y. Liu, and H. Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

[21] 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.

[22] P. Intelligence, B. Ai, A. Amin, R. Aniceto, A. Balakrishna, G. Balke, K. Black, G. Bokinsky, S. Cao, T. Charbonnier, et al. pi0.7: a steerable generalist robotic foundation model with emergent capabilities. *arXiv preprint arXiv:2604.15483*, 2026.

[23] E. Gat, R. P. Bonnasso, R. Murphy, et al. On three-layer architectures. *Artificial Intelligence and Mobile Robots*, 195:210, 1998.

[24] R. C. Arkin. *Behavior-based robotics*. MIT press, 1998.

[25] T. Li, Q. Sun, L. Fan, and K. He. Fractal generative models. *arXiv preprint arXiv:2502.17437*, 2025.

[26] A. Goyal, J. Xu, Y. Guo, V. Blukis, Y.-W. Chao, and D. Fox. Rvt: Robotic view transformer for 3d object manipulation. In *Conference on Robot Learning (CoRL)*, pages 694–710. PMLR, 2023.

[27] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning (CoRL)*, pages 785–799. PMLR, 2023.

[28] X. Ma, S. Patidar, I. Haughton, and S. James. Hierarchical diffusion policy for kinematics-aware multi-task robotic manipulation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 18081–18090, 2024.

[29] Y. Lu, Y. Tian, Z. Yuan, X. Wang, P. Hua, Z. Xue, and H. Xu. H3dp: Triply-hierarchical diffusion policy for visuomotor learning. *arXiv preprint arXiv:2505.07819*, 2025.

[30] Z. Xian and N. Gkanatsios. Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation. In *Conference on Robot Learning (CoRL)*. Proceedings of Machine Learning Research, 2023.

[31] W. Zhang, T. Hu, H. Zhang, Y. Qiao, Y. Qin, Y. Li, J. Liu, T. Kong, L. Liu, and X. Ma. Chain-of-action: Trajectory autoregressive modeling for robotic manipulation. *Advances in Neural Information Processing Systems (NeurIPS)*, 38:106070–106094, 2026.

[32] Q. Bu, H. Li, L. Chen, J. Cai, J. Zeng, H. Cui, M. Yao, and Y. Qiao. Towards synergistic, generalized, and efficient dual-system for robotic manipulation. *arXiv preprint arXiv:2410.08001*, 2024.

[33] C. Cui, P. Ding, W. Song, S. Bai, X. Tong, Z. Ge, R. Suo, W. Zhou, Y. Liu, B. Jia, et al. Openhelix: A short survey, empirical analysis, and open-source dual-system vla model for robotic manipulation. *arXiv preprint arXiv:2505.03912*, 2025.

[34] Z. Gong, P. Ding, S. Lyu, S. Huang, M. Sun, W. Zhao, Z. Fan, and D. Wang. Carp: Visuomotor policy learning via coarse-to-fine autoregressive prediction. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13460–13470, 2025.

[35] K. S. Lashley et al. *The problem of serial order in behavior*, volume 21. Bobbs-Merrill Oxford, 1951.

[36] E. K. Miller and J. D. Cohen. An integrative theory of prefrontal cortex function. *Annual Review of Neuroscience*, 24(1):167–202, 2001.

[37] M. M. Botvinick. Hierarchical models of behavior and prefrontal function. *Trends in cognitive sciences*, 12(5):201–208, 2008.

[38] J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[39] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.

[40] A. D. Vuong, M. N. Vu, and I. Reid. Improving robotic manipulation with efficient geometry-aware vision encoder. *arXiv preprint arXiv:2509.15880*, 2025.

[41] Z. Song, S. Qin, T. Chen, L. Lin, and G. Wang. Physical autoregressive model for robotic manipulation without action pretraining. *arXiv preprint arXiv:2508.09822*, 2025.

[42] O. M. 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.

[43] D. Yashima, K. Seno, S. Kurita, Y. Oda, and K. Sugiura. Hiflow: Tokenization-free scale-wise autoregressive policy learning via flow matching. *arXiv preprint arXiv:2603.27281*, 2026.

[44] T. B. H. R. I. Center. X-humanoid tien kung, 2024. URL https://x-humanoid.com/.

[45] K. Tian, Y. Jiang, Z. Yuan, B. Peng, and L. Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:84839–84865, 2024.

[46] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024.

[47] Y. Ze, Z. Chen, W. Wang, T. Chen, X. He, Y. Yuan, X. B. Peng, and J. Wu. Generalizable humanoid manipulation with 3d diffusion policies. *arXiv preprint arXiv:2410.10803*, 2024.

[48] J. Cao, Q. Zhang, J. Sun, J. Wang, H. Cheng, Y. Li, J. Ma, K. Wu, Z. Xu, Y. Shao, et al. Mamba policy: Towards efficient 3d diffusion policy with hybrid selective state models. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 11359–11366. IEEE, 2025.

[49] J. Cao, Q. Zhang, H. Guo, J. Wang, H. Cheng, and R. Xu. Modality-composable diffusion policy via inference-time distribution-level composition. *arXiv preprint arXiv:2503.12466*, 2025.

[50] J. Cao, Y. Huang, H. Guo, R. Zhang, M. Nan, W. Mai, J. Wang, H. Cheng, J. Sun, G. Han, et al. Compose your policies! improving diffusion-based or flow-based robot policies via test-time distribution-level composition. *arXiv preprint arXiv:2510.01068*, 2025.

[51] Z. Weng, H. Lu, D. Kragic, and J. Lundell. Dexdiffuser: Generating dexterous grasps with diffusion models. *Robotics and Automation Letters (RAL)*, 2024.

[52] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. In *Conference on Robot Learning (CoRL)*, pages 1949–1974. PMLR, 2025.

[53] V. Vosylius, Y. Seo, J. Uruç, and S. James. Render and diffuse: Aligning image and action spaces for diffusion-based behaviour cloning. *arXiv preprint arXiv:2405.18196*, 2024.

[54] T. Z. Zhao, J. Tompson, D. Driess, P. Florence, S. K. S. Ghasemipour, C. Finn, and A. Wahid. Aloha unleashed: A simple recipe for robot dexterity. In *Conference on Robot Learning (CoRL)*, pages 1910–1924. PMLR, 2025.

[55] Q. Yang, M. C. Welle, D. Kragic, and O. Andersson. S2-diffusion: Generalizing from instance-level to category-level skills in robot manipulation. *arXiv preprint arXiv:2502.09389*, 2025.

[56] V. Saxena, Y. Koga, and D. Xu. Constrained-context conditional diffusion models for imitation learning. *arXiv preprint arXiv:2311.01419*, 2023.

[57] S. H. Høeg, Y. Du, and O. Egeland. Streaming diffusion policy: Fast policy synthesis with variable noise diffusion models. *arXiv preprint arXiv:2406.04806*, 2024.

[58] Y. Liu, J. Hamid, A. Xie, Y. Lee, M. Du, and C. Finn. Bidirectional decoding: Improving action chunking via guided test-time sampling. In *International Conference on Learning Representations (ICLR)*, volume 2025, pages 4594–4627, 2025.

[59] X. Li, V. Belagali, J. Shang, and M. S. Ryoo. Crossway diffusion: Improving diffusion-based visuomotor policy via self-supervised learning. In *International Conference on Robotics and Automation (ICRA)*, pages 16841–16849. IEEE, 2024.

[60] D. Wang, S. Hart, D. Surovik, T. Kelestemur, H. Huang, H. Zhao, M. Yeatman, J. Wang, R. Walters, and R. Platt. Equivariant diffusion policy. *arXiv preprint arXiv:2407.01812*, 2024.

[61] X. Zhang, Y. Liu, H. Chang, L. Schramm, and A. Boularias. Autoregressive action sequence learning for robotic manipulation. *Robotics and Automation Letters (RAL)*, 2025.

[62] T. Gervet and Z. Xiao. Act3d: 3d feature field transformers for multi-task robotic manipulation. In *Conference on Robot Learning (CoRL)*. Proceedings of Machine Learning Research, 2023.

[63] Y. Liu, Y. Liu, Y. Meng, J. Zhang, Y. Zhou, Y. Li, J. Jiang, K. Ji, S. Ge, Z. Wang, et al. Spatial policy: Guiding visuomotor robotic manipulation with spatial-aware modeling and reasoning. *arXiv preprint arXiv:2508.15874*, 2025.

[64] G. Yan, J. Zhu, Y. Deng, S. Yang, R.-Z. Qiu, X. Cheng, M. Memmel, R. Krishna, A. Goyal, X. Wang, et al. Maniflow: A general robot manipulation policy via consistency flow training. *arXiv preprint arXiv:2509.01819*, 2025.

[65] D. McAllister, S. Ge, B. Yi, C. M. Kim, E. Weber, H. Choi, H. Feng, and A. Kanazawa. Flow matching policy gradients. *arXiv preprint arXiv:2507.21053*, 2025.

[66] D. Gao, B. Zhao, A. Lee, I. Chuang, H. Zhou, H. Wang, Z. Zhao, J. Zhang, and I. Soltani. Vita: Vision-to-action flow matching policy. *arXiv preprint arXiv:2507.13231*, 2025.

[67] M. Du and S. Song. Dynaguide: Steering diffusion polices with active dynamic guidance. *arXiv preprint arXiv:2506.13922*, 2025.

[68] Z. Sun and S. Song. Latent policy barrier: Learning robust visuomotor policies by staying in-distribution. *arXiv preprint arXiv:2508.05941*, 2025.

[69] M. Reuss, Ö. E. Yağmurlu, F. Wenzel, and R. Lioutikov. Multimodal diffusion transformer: Learning versatile behavior from multimodal goals. *Robotics: Science and Systems (RSS)*, 2024.

[70] W. Liu, T. Hermans, S. Chernova, and C. Paxton. Structdiffusion: Object-centric diffusion for semantic rearrangement of novel objects. In *CoRL Workshop on Language and Robotics*, 2022.

[71] L. Chen, S. Bahl, and D. Pathak. Playfusion: Skill acquisition via diffusion from language-annotated play. In *Conference on Robot Learning (CoRL)*, pages 2012–2029. PMLR, 2023.

[72] T. Yu, T. Xiao, A. Stone, J. Tompson, A. Brohan, S. Wang, J. Singh, C. Tan, J. Peralta, B. Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.

[73] A. Ajay, S. Han, Y. Du, S. Li, A. Gupta, T. Jaakkola, J. Tenenbaum, L. Kaelbling, A. Srivastava, and P. Agrawal. Compositional foundation models for hierarchical planning. *Advances in Neural Information Processing Systems (NeurIPS)*, 36:22304–22325, 2023.

[74] S. Sharan, R. Zhao, Z. Wang, S. P. Chinchali, et al. Plan diffuser: Grounding llm planners with diffusion models for robotic manipulation. In *Bridging the Gap between Cognitive Science and Robot Learning in the Real World: Progresses and New Directions*, 2024.

[75] J. Wen, Y. Zhu, J. Li, M. Zhu, Z. Tang, K. Wu, Z. Xu, N. Liu, R. Cheng, C. Shen, et al. Tinyvla: Towards fast, data-efficient vision-language-action models for robotic manipulation. *Robotics and Automation Letters (RAL)*, 2025.

[76] J. Wen, Y. Zhu, M. Zhu, Z. Tang, J. Li, Z. Zhou, X. Liu, C. Shen, Y. Peng, and F. Feng. Diffusionvla: Scaling robot foundation models via unified diffusion and autoregression. In *International Conference on Machine Learning (ICML)*, 2025.

[77] S. Ye, J. Jang, B. Jeon, S. Joo, J. Yang, B. Peng, A. Mandlekar, R. Tan, Y.-W. Chao, B. Y. Lin, et al. Latent action pretraining from videos. *arXiv preprint arXiv:2410.11758*, 2024.

[78] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning (CoRL)*, pages 2165–2183. PMLR, 2023.

[79] A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.

[80] Z. Zhou, Y. Zhu, M. Zhu, J. Wen, N. Liu, Z. Xu, W. Meng, R. Cheng, Y. Peng, C. Shen, et al. Chatvla: Unified multimodal understanding and robot control with vision-language-action model. *arXiv preprint arXiv:2502.14420*, 2025.

[81] S. Liu, B. Li, K. Ma, L. Wu, H. Tan, X. Ouyang, H. Su, and J. Zhu. Rdt2: Exploring the scaling limit of umi data towards zero-shot cross-embodiment generalization. *arXiv preprint arXiv:2602.03310*, 2026.

[82] J. Liu, H. Chen, P. An, Z. Liu, R. Zhang, C. Gu, X. Li, Z. Guo, S. Chen, M. Liu, et al. Hybridvla: Collaborative diffusion and autoregression in a unified vision-language-action model. *arXiv preprint arXiv:2503.10631*, 2025.

[83] H. Zhen, X. Qiu, P. Chen, J. Yang, X. Yan, Y. Du, Y. Hong, and C. Gan. 3d-vla: a 3d vision-language-action generative world model. In *International Conference on Machine Learning (ICML)*, pages 61229–61245, 2024.

[84] W. Cai, I. Ponomarenko, J. Yuan, X. Li, W. Yang, H. Dong, and B. Zhao. Spatialbot: Precise spatial understanding with vision language models. In *International Conference on Robotics and Automation (ICRA)*, pages 9490–9498. IEEE, 2025.

[85] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

[86] T. Jiang, T. Yuan, Y. Liu, C. Lu, J. Cui, X. Liu, S. Cheng, J. Gao, H. Xu, and H. Zhao. Galaxea open-world dataset and g0 dual-system vla model. *arXiv preprint arXiv:2509.00576*, 2025.

[87] W. Huang, C. Wang, Y. Li, R. Zhang, and L. Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. In *Conference on Robot Learning (CoRL)*, pages 4573–4602. PMLR, 2025.

[88] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. In *Conference on Robot Learning (CoRL)*, pages 540–562. PMLR, 2023.

[89] Y. Du, S. Yang, B. Dai, H. Dai, O. Nachum, J. Tenenbaum, D. Schuurmans, and P. Abbeel. Learning universal policies via text-guided video generation. *Advances in Neural Information Processing Systems (NeurIPS)*, 36:9156–9172, 2023.

[90] H. Bharadhwaj, D. Dwibedi, A. Gupta, S. Tulsiani, C. Doersch, T. Xiao, D. Shah, F. Xia, D. Sadigh, and S. Kirmani. Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation. *arXiv preprint arXiv:2409.16283*, 2024.

[91] H. Wu, Y. Jing, C. Cheang, G. Chen, J. Xu, X. Li, M. Liu, H. Li, and T. Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. In *International Conference on Learning Representations (ICLR)*, volume 2024, pages 10641–10662, 2024.

[92] J. Cen, C. Yu, H. Yuan, Y. Jiang, S. Huang, J. Guo, X. Li, Y. Song, H. Luo, F. Wang, et al. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

[93] J. Cen, S. Huang, Y. Yuan, K. Li, H. Yuan, C. Yu, Y. Jiang, J. Guo, X. Li, H. Luo, et al. Rynnvla-002: A unified vision-language-action and world model. *arXiv preprint arXiv:2511.17502*, 2025.

[94] L. Li, Q. Zhang, Y. Luo, S. Yang, R. Wang, F. Han, M. Yu, Z. Gao, N. Xue, X. Zhu, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026.

[95] C.-L. Cheang, G. Chen, Y. Jing, T. Kong, H. Li, Y. Li, Y. Liu, H. Wu, J. Xu, Y. Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

[96] M. J. Kim, Y. Gao, T.-Y. Lin, Y.-C. Lin, Y. Ge, G. Lam, P. Liang, S. Song, M.-Y. Liu, C. Finn, et al. Cosmos policy: Fine-tuning video models for visuomotor control and planning. *arXiv preprint arXiv:2601.16163*, 2026.

[97] J. Jang, S. Ye, Z. Lin, J. Xiang, J. Bjorck, Y. Fang, F. Hu, S. Huang, K. Kundalia, Y.-C. Lin, et al. Dreamgen: Unlocking generalization in robot learning through video world models. *arXiv preprint arXiv:2505.12705*, 2025.

[98] Y. Liao, P. Zhou, S. Huang, D. Yang, S. Chen, Y. Jiang, Y. Hu, J. Cai, S. Liu, J. Luo, et al. Genie envisioner: A unified world foundation platform for robotic manipulation. *arXiv preprint arXiv:2508.05635*, 2025.

[99] S. Ye, Y. Ge, K. Zheng, S. Gao, S. Yu, G. Kurian, S. Indupuru, Y. L. Tan, C. Zhu, J. Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

[100] S. Gao, W. Liang, K. Zheng, A. Malik, S. Ye, S. Yu, W.-C. Tseng, Y. Dong, K. Mo, C.-H. Lin, et al. Dreamdojo: A generalist robot world model from large-scale human videos. *arXiv preprint arXiv:2602.06949*, 2026.

[101] A. Ye, B. Wang, C. Ni, G. Huang, G. Zhao, H. Li, H. Li, J. Li, J. Lv, J. Liu, et al. Gigaworld-policy: An efficient action-centered world–action model. *arXiv preprint arXiv:2603.17240*, 2026.

[102] J. Won, K. Lee, H. Jang, D. Kim, and J. Shin. Dual-stream diffusion for world-model augmented vision-language-action model. *arXiv preprint arXiv:2510.27607*, 2025.

[103] H. Bi, H. Tan, S. Xie, Z. Wang, S. Huang, H. Liu, R. Zhao, Y. Feng, C. Xiang, Y. Rong, et al. Motus: A unified latent action world model. *arXiv preprint arXiv:2512.13030*, 2025.

[104] M. Team, C. Xiang, F. Bao, H. Liu, H. Tan, H. Bi, J. Li, J. Liu, J. Pang, K. Jing, et al. Motubrain: An advanced world action model for robot control. *arXiv preprint arXiv:2604.27792*, 2026.

[105] S. Zhou, Y. Du, J. Chen, Y. Li, D.-Y. Yeung, and C. Gan. Robodreamer: Learning compositional world models for robot imagination. *arXiv preprint arXiv:2404.12377*, 2024.

[106] R. Zheng, J. Wang, S. Reed, J. Bjorck, Y. Fang, F. Hu, J. Jang, K. Kundalia, Z. Lin, L. Magne, et al. Flare: Robot learning with implicit world modeling. *arXiv preprint arXiv:2505.15659*, 2025.

[107] Y. Feng, H. Tan, X. Mao, C. Xiang, G. Liu, S. Huang, H. Su, and J. Zhu. Vidar: Embodied video diffusion model for generalist manipulation. *arXiv preprint arXiv:2507.12898*, 2025.

[108] W. Zhang, H. Liu, Z. Qi, Y. Wang, X. Yu, J. Zhang, R. Dong, J. He, H. Wang, Z. Zhang, et al. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *Advances in Neural Information Processing Systems (NeurIPS)*, 38:24195–24228, 2026.

[109] P. Zhou, L. Chen, S. Chen, D. Chen, W. Zhao, R. Jin, G. Ren, and J. Luo. Act2goal: From world model to general goal-conditioned policy. *arXiv preprint arXiv:2512.23541*, 2025.

[110] J. Lyu, K. Liu, X. Zhang, H. Liao, Y. Feng, W. Zhu, T. Shen, J. Chen, J. Zhang, Y. Dong, et al. Lda-1b: Scaling latent dynamics action model via universal embodied data ingestion. *arXiv preprint arXiv:2602.12215*, 2026.

[111] C. Zhu, R. Yu, S. Feng, B. Burchfiel, P. Shah, and A. Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. *arXiv preprint arXiv:2504.02792*, 2025.

[112] Q. Zhao, Y. Lu, M. J. Kim, Z. Fu, Z. Zhang, Y. Wu, Z. Li, Q. Ma, S. Han, C. Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1702–1713, 2025.

Appendix Overview

The appendix provides supplementary materials, extended analyses, and comprehensive implementation details to support the main text. It is structured as follows:

1. **Detailed Experiment Settings** – Comprehensive configurations for both simulations and real-world robotic deployments, including hardware specifications and hyperparameter setups.
2. **Real-world Robot Settings** – Detailed robot and task settings for real-world deployment.
3. **Efficiency Testing** – Quantitative benchmarking of computational footprint and inference latency, comparing our RCP against established one-pass monolithic architectures.
4. **Detailed Real-World Trial-Level Results**: A transparent and granular breakdown of raw execution counts and empirical success rates across all physical deployments on the dual-arm platform.
5. **Extended Multi-Scale Policy Comparison**: A rigorous architectural analysis and empirical benchmark contrasting RCP with the latest multi-scale generation paradigms, specifically CARP and HiFlow.
6. **Algorithmic Implementation Details** – Complete algorithmic pseudo-code delineating the dual-mode training procedures (segment mode and cascade mode) and the recursive inference pipeline.
7. **Comprehensive Visualizations** – Extended sequential execution rollouts across all simulated benchmarks and physical manipulation tasks.
8. **Detailed Literature Review** – An expanded discussion focused on the latest vision-action (VA) models, vision-language-action (VLA) models, and world-action-model (WAM).

A Experiment Settings

A.1 Simulation Experiments

Base Settings. We train on expert demonstrations and evaluate with closed-loop rollouts in the simulator. Actions are joint positions (qpos) with task-specific dimensionality. Training mixes segment mode and cascade mode supervision with a frequency ratio of 0.8. (All experiments use a ratio of 0.8 unless noted elsewhere) We use AdamW (β_1=0.9, β_2=0.95), cosine learning-rate decay with 40 warmup epochs, gradient clipping at 3.0, and L1 losses on joint and gripper dimensions. Actions are z-score normalized using training-set statistics. At deployment, we apply ACT-style temporal ensembling with decay 0.01 over overlapping 16-step prediction chunks.

RoboTwin. We evaluate on RoboTwin 2.0. The robot is an ALOHA-style dual-arm setup with a 14-dimensional action vector: left arm (6) + left gripper (1) + right arm (6) + right gripper (1). Observations consist of a single head RGB camera at 224×224 (ResNet-18 backbone) and current proprioception. Raw RoboTwin HDF5 episodes are converted to our training format by concatenating joint actions and decoding JPEG camera frames. We use 50 expert demonstrations per task unless stated otherwise.

We adopt the `base` architecture preset: tree depth L=2, branching factor k=4, output horizon k^L=16, embedding dimension 256, 8 attention heads, and 6 transformer blocks. Training runs for $20{,}000$ epochs with batch size 64, learning rate 10^{-4}, weight decay 0.05, and BatchNorm freezing enabled. Evaluation uses the official RoboTwin `eval_policy.py` script with `instruction_type=unseen`. We report success rate over 100 valid test episodes per task. For the point-cloud RCP variant, we replace RGB with 128 xyz point-cloud tokens (3D input, centered coordinates) and train with batch size 50 on the same 50-demo protocol across three manipulation tasks.

MainSkill. We additionally benchmark on three ManiSkill tasks: `PickCube-v1`, `PushCube-v1`, and `StackCube-v1`. For each task we collect 100 expert trajectories and train one task-specific policy with the same `base` configuration as above (L=2, k=4, 16 waypoints, embed dim 256, 8 heads, 6 blocks). We use random-segment ratio as 0.6, 0.5, and 0.5, respectively. All other training hyperparameters match the default RCP settings in Sec. A.1. Evaluation follows the standard ManiSkill success criterion for each task.

Ablation on Model Scaling. We study model capacity on three RoboTwin tasks: `pick_diverse_bottles`, `put_object_cabinet`, and `place_container_plate`. We compare three presets while keeping the image observation pipeline, optimizer, and training schedule fixed:

Preset	L	Horizon (k^L)	Embed dim	Heads	Blocks	Param. (M)
`small`	2	16	128	4	4	12.84
`base`	2	16	256	8	6	19.76
`large`	3	64	512	8	12	70.34

Table A.1: Model presets for the scaling ablation. All presets use k=4 and a bidirectional generator.

Training uses 20,000 epochs, seed 42, learning rate 10^{-4}, backbone learning rate 10^{-5}, random-segment ratio 0.8, breadth-first batch size 8, and z-score action normalization. Batch sizes are 50 to fit GPU memory. Evaluation follows the RoboTwin protocol above (100 episodes, unseen instructions).

A.2 Real-World Experiments

We evaluate on four tasks collected on a Tienkung dual-arm platform (station 29): finding packaging tape and placing it in a basket, opening a white solid-wood drawer, pick-and-weigh, and stack-bowl. Demonstrations are stored as per-episode HDF5 files under a shared data root.

Data processing. Each trajectory is stored as `trajectory.hdf5` with aligned puppet joint streams and JPEG-compressed RGB frames from `camera_top`, `camera_left`, and `camera_right`. Trajectories are sorted by ID. We use the first 100 trajectories for training and trajectories 100–109 for validation. Single-arm tasks (drawer opening) use an 8-DoF action vector: 7 arm joints plus 1 gripper value. Bimanual tasks use 16 DoF in the order [left arm (7), left gripper (1), right arm (7), right gripper (1)]. States and actions are z-score normalized on the training split (σ clipped at 10^{-3}). Images are decoded from JPEG, converted to RGB, and resized with area interpolation. For our RCP, raw HDF5 files are preprocessed once into RCP-native episodes containing `/trajectory`, `/qpos_init`, and `/images/camera_top` at 224×224.

Training settings. All image policies share a unified protocol unless noted: prediction horizon H=16, batch size 128, 100,000 optimizer steps, seed 42, 4 dataloader workers, and checkpoint milestones at 30k, 50k, 80k, and 100k steps. Validation is capped at 20 batches per epoch; `best` checkpoints are selected by lowest validation loss on trajectories 100–109.

- **ACT**
  - **Cameras:** top, left, right
  - **Image size:** 224^2
  - **Key hyperparameters:** ResNet-18; hidden 512; 4+7 enc/dec layers; chunk 16; lr 3×10^{-4}; backbone lr 10^{-5}; KL weight 10
  - **Optimizer:** Unified AdamW (wd 10^{-4}, grad clip 10)
- **Diffusion Policy**
  - **Cameras:** top (2 obs steps)

  - **Image size:** 224^2
  - **Key hyperparameters:** ResNet-18 UNet; DDPM 100 steps; down dims $(256, 512, 1024)$; lr 3×10^{-4}
  - **Optimizer:** Unified AdamW (wd 10^{-4}, grad clip 10)
- **CARP**
  - **Cameras:** top + right wrist
  - **Image size:** 84^2
  - **Key hyperparameters:** ResNet-18 per camera; VAE trained 10,000 steps; AR depth 16, embed 160; lr 10^{-4} (AR), 3×10^{-4} (VAE)
  - **Optimizer:** Unified AdamW (wd 10^{-4}, grad clip 10)
- **Recursive Cascade Policy**
  - **Cameras:** top
  - **Image size:** 224^2
  - **Key hyperparameters:** `base` RCP architecture; lr 10^{-4}, wd 0.05; cosine schedule; 100% random-segment; oversample $\times20$ (≈ 6667 epochs)
  - **Optimizer:** Native training loop with grad clip 3 and mixed precision

ACT samples chunk mode ($qpos_t \rightarrow qpos_{t+1:t+H}$). DP and CARP use sequence mode with $H+1$ frames. RCP sets task-specific `action_dim` (8 or 16) and gripper indices accordingly.

B Real-World Robot and Task Settings

To evaluate the physical efficacy and spatial grounding capabilities of our framework, we deploy the model on a physical hardware platform featuring the TienKung 2.0 [44] dual-arm robot. As illustrated in Figure 8, the robotic system is equipped with two 7-DoF manipulators, each fitted with a Robotiq parallel gripper, providing the dexterity required for complex physical interactions.

Our real-world evaluation suite is explicitly designed to stress-test the policy across varying degrees of manipulation complexity. It comprises four distinct tasks, categorized by their reliance on bimanual coordination versus fine single-arm execution:

- **Tape Transfer** (`find_out_packaging_tape_into_the_other_basket`): The robot must extract packaging tape from a cluttered basket and precisely transport it to a designated target receptacle. This task heavily relies on spatial awareness and bimanual coordination to navigate structural clutter.
- **Weighing Operation** (`pick_up_weigh`): A multi-stage sequential task where the robot first positions a basket onto a scale and subsequently places a target object inside. It explicitly evaluates the model's capacity for long-horizon planning and synchronous bimanual execution.
- **Bowl Stacking** (`stack_bowl`): The robot utilizes both arms collaboratively to grasp, align, and stack bowls. This tests the policy's robustness in maintaining precise relative state estimation during close-proximity bimanual interaction.
- **Drawer Opening** (`open_white_solid_wood_drawers`): In contrast to the bimanual tasks, this evaluates high-precision single-arm manipulation. The robot must accurately grasp the handle and pull open a solid wood drawer, demanding exact spatial grounding and fine local motor refinement.

C Efficiency Testing

Experiment Settings. We benchmark end-to-end inference latency on a single NVIDIA A100-SXM4-80GB GPU with batch size one. All policies use randomly initialized weights and fixed

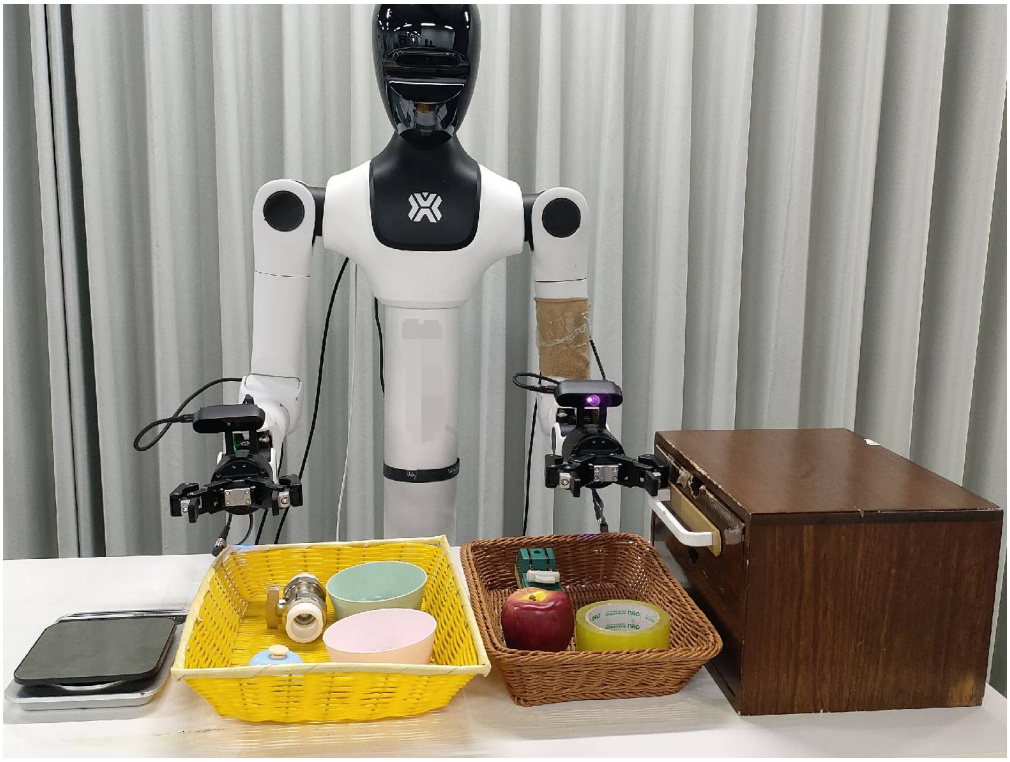

Figure 8: **Real-world experimental setup and task suite.** The TienKung 2.0 dual-arm robot executing diverse manipulation tasks. The suite comprehensively evaluates both synchronous bimanual coordination and high-precision single-arm motor control.

Table B.2: Inference efficiency across action horizons H. Latency is the median wall-clock time per chunk.

Method	H	Params (M)	ms/chunk	ms/action	Hz/action	Hz/chunk
RCP (ours)	4	19.8	11.7	2.93	342	85.4
	16	19.8	18.1	1.13	886	55.4
	64	19.8	24.6	0.38	2601	40.6
ACT [2]	4	106.2	63.3	15.8	63	15.8
	16	106.2	64.1	4.01	250	15.6
	64	106.2	64.5	1.01	992	15.5
DP [1]	4	91.5	2155	539	1.9	0.5
	16	91.5	2172	136	7.4	0.5
	64	91.5	2174	34.0	29.4	0.5
CARP [34]	4	30.2	102	25.5	39	9.8
	16	30.2	106	6.61	151	9.5
	64	30.3	117	1.83	547	8.5

dummy observations, executed through each method's native deployment path in its own conda environment. We report the median wall-clock time over 200 timed forward passes after 50 warmup iterations. Action horizons are $H \in \{4, 16, 64\}$. For RCP we set branching factor k=4 and map H to inference depth $L \in \{1, 2, 3\}$ via $H = k^L$, using a shared generator (d=256, 8 heads, 6 blocks; ≈19.8 M parameters). ACT follows the image policy in our codebase (ResNet-18 backbone per camera, three cameras, transformer encoder depth 4 and decoder depth 7, hidden size 512, feed-forward width 3200; ≈106 M parameters). Diffusion Policy uses a ResNet-18 conditional U-Net with 100 denoising steps (≈91.5 M parameters). CARP uses depth-16 autoregressive stages with embedding width 160 and a dual ResNet-18 observation encoder (≈30.3 M parameters

at $H{=}64$). Latency is measured per *chunk* (one policy call producing H actions). We also report amortized latency per action ($\mathrm{ms/action} = \mathrm{ms/chunk}/H$), sustained action throughput per replan ($\mathrm{Hz/action} = 1000/\mathrm{ms/action}$), and replan rate ($\mathrm{Hz/chunk} = 1000/\mathrm{ms/chunk}$).

Results. Tab. B.2 summarizes the benchmark. RCP keeps a fixed parameter budget while chunk latency grows sublinearly with H (11.7 ms at $H{=}4$ to 24.6 ms at $H{=}64$), because inference cost scales with depth L rather than horizon length directly. Amortized cost per action therefore falls sharply: 2.93 ms/action at $H{=}4$ versus 0.38 ms/action at $H{=}64$ (341 versus 2601 Hz/action). Chunk throughput remains high throughout (85.4 down to 40.6 Hz/chunk).

ACT exhibits near-constant chunk latency ($\approx$64 ms across all H), yielding 15.5 Hz/chunk but improving amortized action cost only through longer horizons (992 Hz/action at $H{=}64$). At $H{=}64$, RCP is $2.6\times$ faster per chunk than ACT while using roughly $5\times$ fewer parameters. Diffusion Policy is dominated by iterative denoising: latency stays near 2.15 s per chunk ($\approx$0.5 Hz/chunk), roughly $88\times$ slower than RCP at $H{=}64$ despite comparable model size. CARP carries a modest parameter count ($\approx$30 M) yet remains $4.8\times$ slower than RCP at $H{=}64$ (117 versus 24.6 ms/chunk), reflecting sequential autoregressive stages and observation encoding rather than total parameter count alone. Overall, RCP offers the most favorable trade-off among the baselines: sublinear growth in chunk latency, strong amortized action throughput at long horizons, and the smallest model footprint.

D Detailed Real-World Trial-Level Results

To provide full transparency into the physical robot evaluations, we report the exact success-to-trial counts for all 80 deployment runs on the Tien Kung 2.0 Humaniod platform in Tab. D.3. Each of the four tasks was subjected to 20 independent evaluation trials under varying object placements and initial configurations.

Across the synchronous bimanual coordination tasks, RCP establishes clear performance margins over all baselines. Specifically, in the `Stack Bowl` task, RCP achieves a 50% success rate (10/20), outperforming the strongest monolithic baseline by an absolute margin of 15%. For the highly dynamic `Transfer Tape` task, RCP reaches a peak success rate of 65% (13/20), whereas CARP drops significantly to 15% (3/20) due to execution failures stemming from discrete quantization. In the sequential, multi-stage `Weigh Apple` task, RCP maintains a 35% success rate (7/20), leading the nearest baseline by 10%.

For the high-precision articulation category, represented by the `Open Drawer` task, RCP secures the highest execution fidelity with an 80% success rate (16/20), while ACT and Diffusion Policy achieve 75% and 50% respectively. Cumulatively, across the entire physical task suite, RCP delivers a definitive overall success rate of 58% (46/80), reinforcing its robust capability to ground multi-scale policy learning into reliable real-world motor execution.

Table D.3: **Complete Quantitative Results and Raw Trial Counts on the Real-World Robot.**

	Bimanual Coordination Tasks			**High-Precision Task**	
Method	`Stack Bowl`	`Transfer Tape`	`Weigh Apple`	`Open Drawer`	**Overall**
ACT	6/20 (30%)	6/20 (30%)	4/20 (20%)	15/20 (75%)	31/80 (39%)
Diffusion Policy	4/20 (20%)	6/20 (30%)	5/20 (25%)	10/20 (50%)	25/80 (31%)
CARP	7/20 (35%)	3/20 (15%)	3/20 (15%)	12/20 (60%)	25/80 (31%)
RCP (Ours)	**10/20 (50%)**	**13/20 (65%)**	**7/20 (35%)**	**16/20 (80%)**	**46/80 (58%)**

E Extended Comparison with Multi-Scale Action Policies

While the fundamental motivation and generative mechanisms of our Recursive Cascade Policy (RCP) differ significantly from recent Visual Autoregressive (VAR [45]) adaptations in robotics,

Table E.4: **Quantitative Comparison with Multi-Scale Policies in RoboTwin.** Success rates evaluated across complex manipulation tasks. RCP demonstrates a decisive and absolute advantage over both discrete (CARP) and continuous (HiFlow) multi-scale baselines. Entries marked with N/A indicate unavailable data due to closed-source constraints.

RoboTwin Task	CARP	HiFlow	RCP (Ours)
beat_block_hammer	0.08	N/A	**0.81**
pick_diverse_bottles	0.00	N/A	**0.24**
place_phone_stand	0.06	N/A	**0.42**
move_can_pot	0.04	0.42	**0.54**
click_alarm	0.68	0.69	**0.76**
place_can_basket	0.00	0.39	**0.57**

they share a conceptual alignment in addressing multi-scale action generation. To rigorously contextualize our contributions, we provide a detailed architectural contrast and quantitative benchmarking against the latest state-of-the-art multi-scale policies: CARP [34] and HiFlow [43].

Architectural and Mechanistic Contrast. CARP [34] advances the coarse-to-fine paradigm by compressing actions through a multi-stage VQ-VAE and modeling the latent sequence autoregressively. However, this approach introduces several critical limitations. First, operating within a quantized discrete space inherently risks quantization artifacts, stripping away the continuous physical inductive biases crucial for high-fidelity motor control. Second, generating finer scales in CARP requires an expanding attention context that strictly encompasses all preceding coarser scales, leading to computational bottlenecks. Finally, the framework mandates a fragile two-stage training pipeline where the action VAE must be trained and frozen prior to optimizing the autoregressive backbone.

More recently, HiFlow [43] attempts to rectify the two-stage training bottleneck by adopting an end-to-end continuous flow-matching objective. Despite this architectural shift, it remains inherently tethered to the expanding multi-scale token formulation. In stark contrast, RCP operates entirely in an explicit continuous space, persisting state via a single continuous latent representation. By employing a shared recursive generator, RCP explicitly supervises continuous action outputs at every cascade level without scaling the attention context. This elegant formulation ensures a strict correspondence with physical reality while completely bypassing the compounding errors typical of multi-stage quantization pipelines.

Empirical Evaluation in RoboTwin. To substantiate our architectural advantages, we conduct rigorous comparative evaluations in the highly dynamic RoboTwin simulation environment. We structure our evaluation into two distinct task sets to ensure comprehensive and fair benchmarking. For the first set, we randomly select three demanding manipulation tasks (`beat_block_hammer`, `pick_diverse_bottles`, and `place_phone_stand`) to directly contrast RCP against CARP. For the second set, we specifically evaluate RCP against both CARP and HiFlow on three identical tasks reported in the recent HiFlow literature (`move_can_pot`, `click_alarm`, and `place_can_basket`). This targeted selection ensures a rigorous comparison given the closed-source nature of the HiFlow codebase.

As summarized in Tab. E.4, RCP establishes an absolute performance superiority across all evaluated dimensions. In the first task set, CARP struggles to ground its discrete latents into successful physical executions, yielding near-zero success rates. Conversely, RCP achieves a success rate of 0.81 on the highly dynamic `beat_block_hammer` task. In the HiFlow-benchmarked tasks, RCP consistently dominates both baselines, achieving a 0.76 success rate on `click_alarm` and 0.57 on `place_can_basket`, significantly outperforming the flow-matching adaptations.

Algorithm 1 Segment Sampling Training

Require: Demonstrations $\mathcal{D}$, shared generator G_θ, observation encoder E_ϕ, waypoint encoder P_ψ, waypoint decoder D_φ

Require: Tree depth L, branching factor k, open-end dropout probability $p_\circ$

for each mini-batch $\mathcal{B}$ **do**

  for each sample in $\mathcal{B}$ **do**

    Sample a random window of k^L consecutive waypoints from a demonstration

    Build trajectory tree; uniformly sample level ℓ and segment s

    Extract boundaries $(\mathbf{t}_{\texttt{start}}, \mathbf{t}_{\texttt{end}})$ and k ground-truth waypoints $\mathcal{A}_s^l = \{\mathbf{t}_{\texttt{start}}, a_2, a_3, ..., a_k\}$

  end for

  $\mathbf{M} \leftarrow E_\phi(\mathbf{o})$ ▷ encode observation → memory tokens

  $\mathbf{z}_{\texttt{start}} \leftarrow P_\psi(\mathbf{t}_{\texttt{start}})$ ▷ embed start boundary

  $\mathbf{z}_{\texttt{end}} \leftarrow \begin{cases} \mathbf{z}_\circ & \text{if segment is open-ended} \\ \mathbf{z}_\circ & \text{if segment is bounded, with prob. } p_\circ \\ P_\psi(\mathbf{t}_{\texttt{end}}) & \text{otherwise} \end{cases}$ ▷ open-end dropout

  $\mathbf{e}_\ell \leftarrow \ell$ ▷ extract level embedding

  $\mathbf{F} \leftarrow G_\theta(\mathbf{z}_{\texttt{start}}, \mathbf{z}_{\texttt{end}}, \mathbf{e}_\ell, \mathbf{M}, \varnothing)$ ▷ k output features

  $\hat{\mathbf{Y}} \leftarrow D_\varphi(\mathbf{F})$ ▷ decode k waypoints

  Update θ via $\nabla_\theta \mathcal{L}\left(\hat{\mathbf{Y}}, \mathcal{A}_s^l\right)$

end for

F Algorithm

To provide a concrete blueprint for implementation and ensure full reproducibility, we present the formal algorithmic procedures of our framework in this section. The training phase is governed by two complementary regimes, detailed as follows: (a) the *Segment Sampling Mode* in Alg. 1, which enforces local precision via teacher-forced boundaries, and (b) the *Cascaded Refinement Mode* in Alg. 2, which explicitly mitigates inter-layer compounding errors by unrolling the hierarchical action tree. Conversely, Alg. 3 outlines the deployment phase, illustrating how the shared recursive generator dynamically unrolls the action manifold to instantiate hierarchical action sequences during closed-loop evaluation.

Algorithm 2 Cascaded Refinement Training

Require: Demonstrations $\mathcal{D}$, shared generator G_θ, observation encoder E_ϕ, waypoint encoder P_ψ, waypoint decoder D_φ, latent aggregator AGG

Require: Tree depth L, branching factor k, per-level loss weights $\{\lambda_\ell\}_{\ell=0}^{L-1}$

for each mini-batch $\mathcal{B}$ **do**
  $\mathbf{M} \leftarrow E_\phi(\mathbf{o})$ ▷ encode observation once
  Initialise root: $\mathcal{S}_0 \leftarrow \big\{(\mathbf{t}_{\texttt{start}} = \mathbf{q}_0,\ \mathbf{z}_{\texttt{end}} = \mathbf{z}_\circ)\big\}$, $\mathbf{h} \leftarrow \varnothing$ ▷ open-ended, no parent latent
  for $\ell = 0, \dots, L-1$ **do**
    $\mathcal{L}_\ell \leftarrow 0$; $\mathcal{S}_{\ell+1} \leftarrow \emptyset$
    $\mathbf{e}_\ell \leftarrow \ell$ ▷ extract level embedding
    for each segment $(\hat{\mathbf{t}}_{\texttt{start}}, \hat{\mathbf{t}}_{\texttt{end}}) \in \mathcal{S}_\ell$ **do** ▷ boundaries are model predictions
      $\mathbf{z}_{\texttt{start}} \leftarrow P_\psi(\hat{\mathbf{t}}_{\texttt{start}})$; $\mathbf{z}_{\texttt{end}} \leftarrow P_\psi(\hat{\mathbf{t}}_{\texttt{end}})$ or $\mathbf{z}_\circ$ if open-ended
      $\mathbf{F} \leftarrow G_\theta\big(\mathbf{z}_{\texttt{start}},\ \mathbf{z}_{\texttt{end}},\ \mathbf{e}_\ell,\ \mathbf{M},\ \mathbf{h}\big)$
      $\hat{\mathbf{Y}} \leftarrow D_\varphi(\mathbf{F})$ ▷ decode k waypoints: $\{\hat{a}_1, \dots, \hat{a}_k\}$
      $\mathcal{L}_\ell \mathrel{+}= \mathcal{L}\Big(\hat{\mathbf{Y}},\ \mathcal{A}_s^\ell\Big)$ ▷ supervise against ground truth
      $\mathcal{S}_{\ell+1} \leftarrow \mathcal{S}_{\ell+1} \cup \textsc{Split}(\hat{\mathbf{Y}},\ \hat{\mathbf{t}}_{\texttt{end}})$ ▷ predicted waypoints → child segments
      $\mathbf{h} \leftarrow \textsc{Agg}(\mathbf{F})$ ▷ aggregate latent for children
    end for
  end for
  Update θ via $\nabla_\theta \sum_{\ell=0}^{L-1} \lambda_\ell\, \mathcal{L}_\ell$
end for

Algorithm 3 Parallelized Inference

Require: Observation $\mathbf{o}$, current joint configuration $\mathbf{q}_0 \in \mathbb{R}^d$, tree depth L, branching factor k

Ensure: Trajectory $\boldsymbol{\tau} = (\hat{\mathbf{q}}_1, \dots, \hat{\mathbf{q}}_{k^L}) \in \mathbb{R}^{k^L \times d}$

$\mathbf{M} \leftarrow E_\phi(\mathbf{o})$ ▷ encode observation once
$\mathbf{T}_s \leftarrow [\mathbf{q}_0]$ ▷ segment start waypoints, length N
$\mathbf{T}_e \leftarrow [\mathbf{q}_0]$ ▷ segment end waypoints, length N
$\mathbf{b} \leftarrow [\textsc{true}]$ ▷ per-segment open-ended flag, length N
$\mathbf{H} \leftarrow \varnothing$ ▷ carried latent (none at root)
for $\ell = 0, \dots, L-1$ **do**
  $N \leftarrow |\mathbf{T}_s|$ ▷ $N = k^\ell$ segments at this level
  $\mathbf{e}_\ell \leftarrow \ell$ ▷ level embedding
  $\mathbf{Z}_s \leftarrow P_\psi(\mathbf{T}_s)$ ▷ embed start boundaries
  for $i = 0, \dots, N-1$ **do** ▷ embed end boundaries with open-end masking
    $\mathbf{Z}_e[i] \leftarrow \begin{cases} \mathbf{z}_\circ & \text{if } \mathbf{b}[i] = \textsc{true} \\ P_\psi(\mathbf{T}_e[i]) & \text{otherwise} \end{cases}$
  end for
  $\mathbf{F} \leftarrow G_\theta(\mathbf{Z}_s,\ \mathbf{Z}_e,\ \mathbf{e}_\ell,\ \mathbf{M}^{\otimes N},\ \mathbf{H})$ ▷ batched generator: (N, k, D)
  $\hat{\mathbf{Y}} \leftarrow D_\varphi(\mathbf{F})$ ▷ decode to (N, k, d): enriched start + $k{-}1$ intermediates
  $\mathbf{H} \leftarrow \textsc{Agg}(\mathbf{F}).\textsc{Broadcast}(k)$ ▷ parent latent → all k children
  ▷ Rearrange N segments × k waypoints into Nk child segments:
  $\mathbf{T}_s \leftarrow \hat{\mathbf{Y}}.\textsc{Reshape}(Nk,\ d)$ ▷ each predicted waypoint becomes a child's start
  for $j = 0, \dots, N-1$ **do** ▷ assign child end boundaries and open-ended flags
    for $i = 0, \dots, k-1$ **do**
      if $i < k-1$ **then** ▷ bounded child: end = next sibling
        $\mathbf{T}_e[jk+i] \leftarrow \hat{\mathbf{Y}}[j,\ i+1]$; $\mathbf{b}[jk+i] \leftarrow \textsc{false}$
      else ▷ last child: inherit parent's end and open-ended flag
        $\mathbf{T}_e[jk+i] \leftarrow \mathbf{T}_e[j]$; $\mathbf{b}[jk+i] \leftarrow \mathbf{b}[j]$
      end if
    end for
  end for
end for
return $\mathbf{T}_s$ ▷ k^L trajectory waypoints, generated in L forward passes

G Comprehensive Visualizations

In this section, we provide extensive visual documentation of our Recursive Cascade Policy in action. To empirically substantiate the quantitative results presented in the main text, we visualize the sequential rollouts across the complete suite of evaluated tasks.

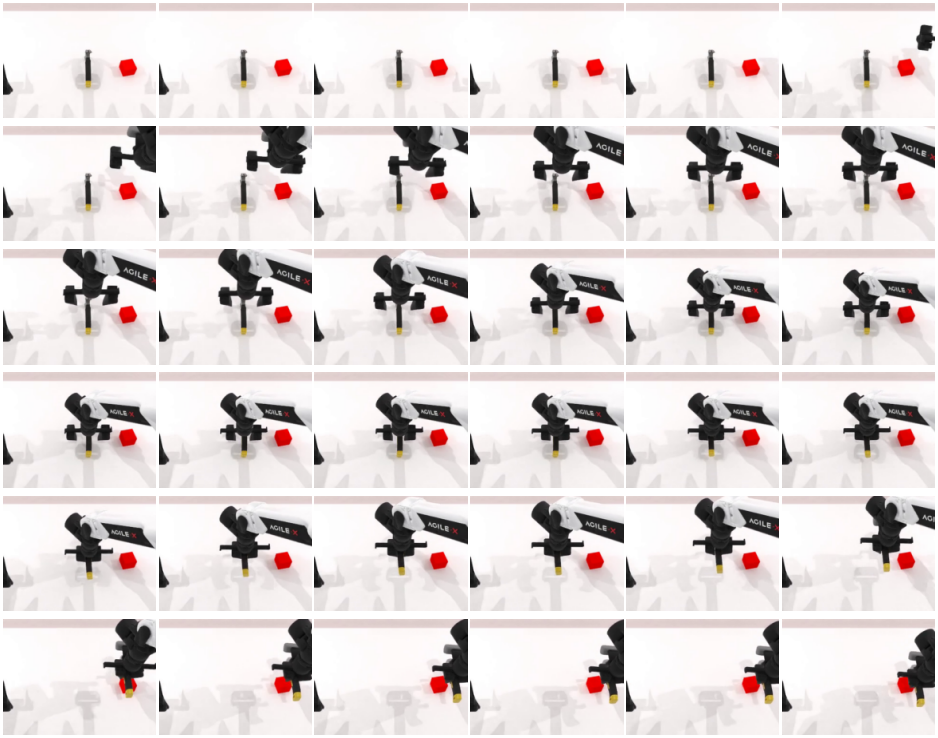

Figure 9: Illustration of `beat_block_hammer` on RoboTwin

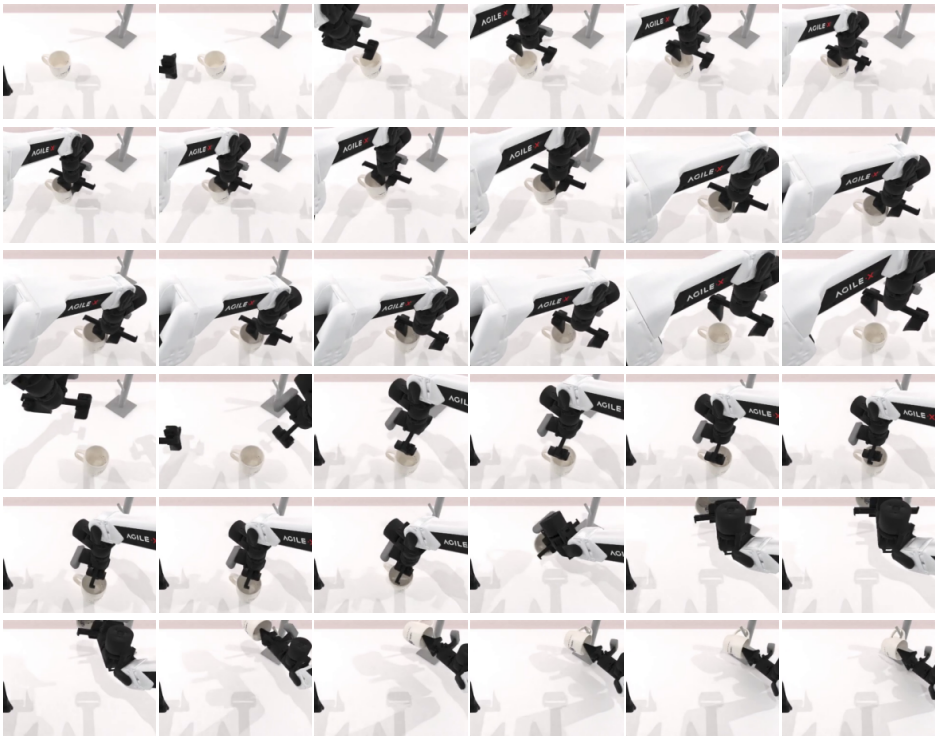

Figure 10: Illustration of `hanging_mug` on RoboTwin

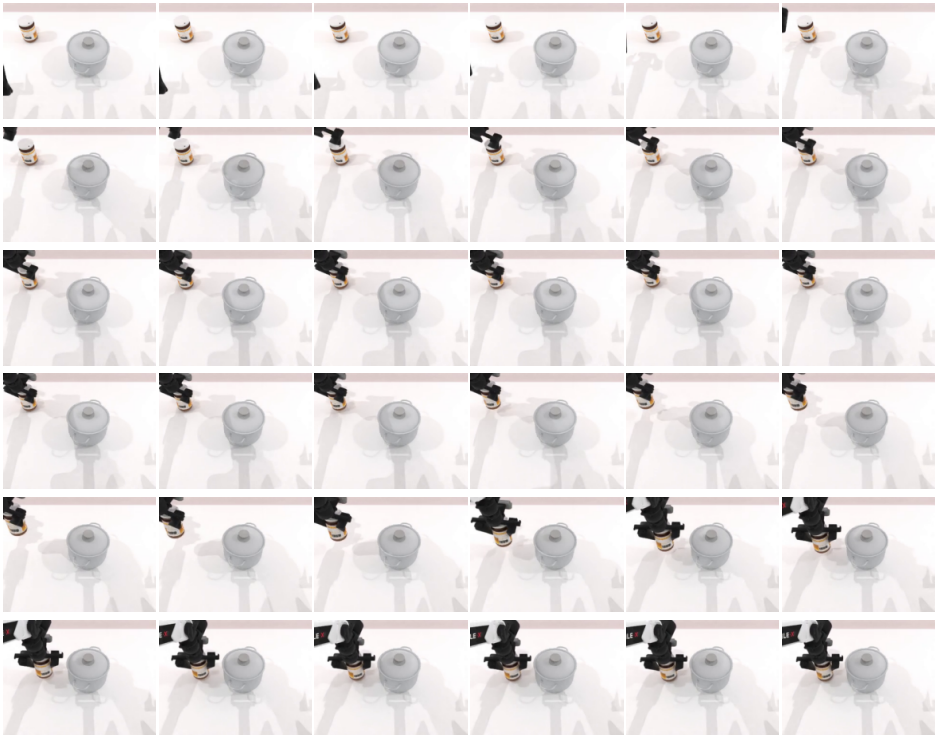

Figure 11: Illustration of `move_can_pot` on RoboTwin

Figure 12: Illustration of `pick_diverse_bottles` on RoboTwin

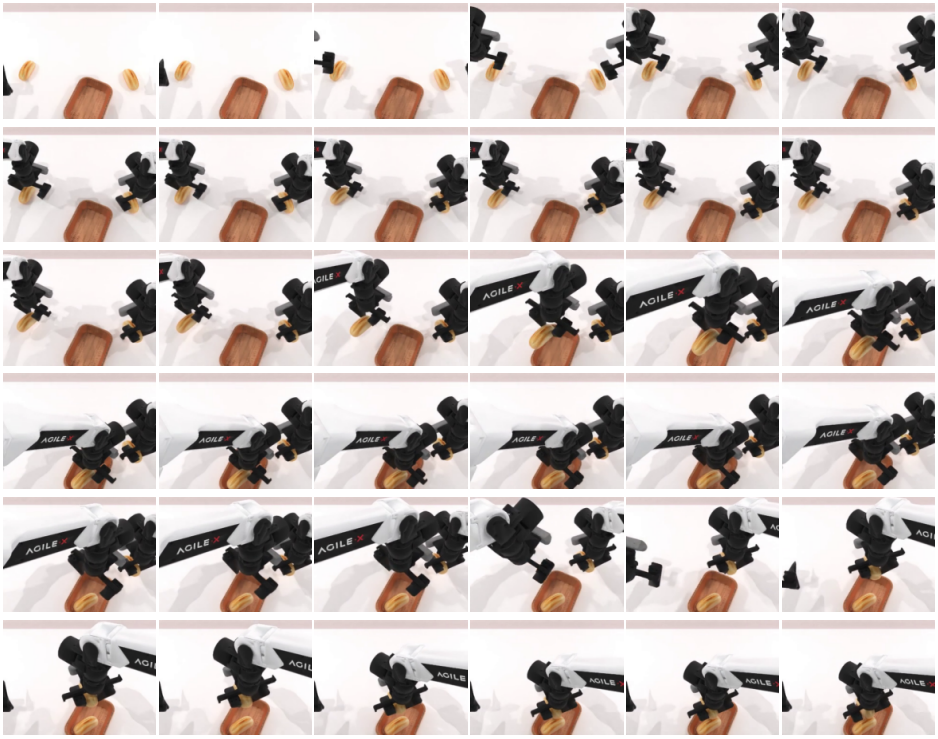

Figure 13: Illustration of `place_bread_basket` on RoboTwin

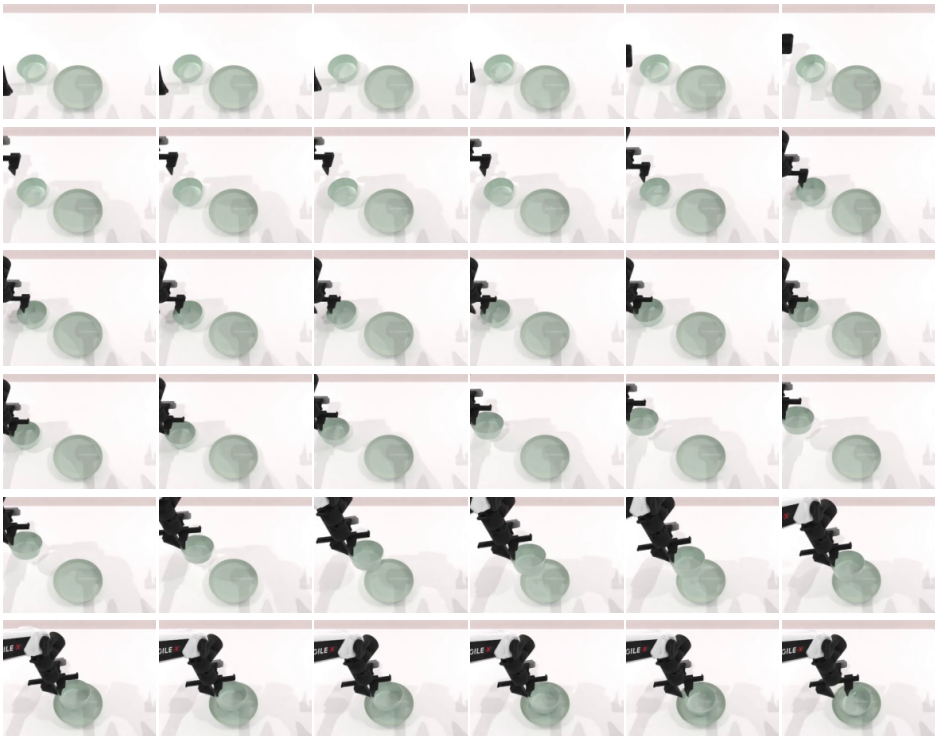

Figure 14: Illustration of `place_container_plate` on RoboTwin

Figure 15: Illustration of `place_dual_shoes` on RoboTwin

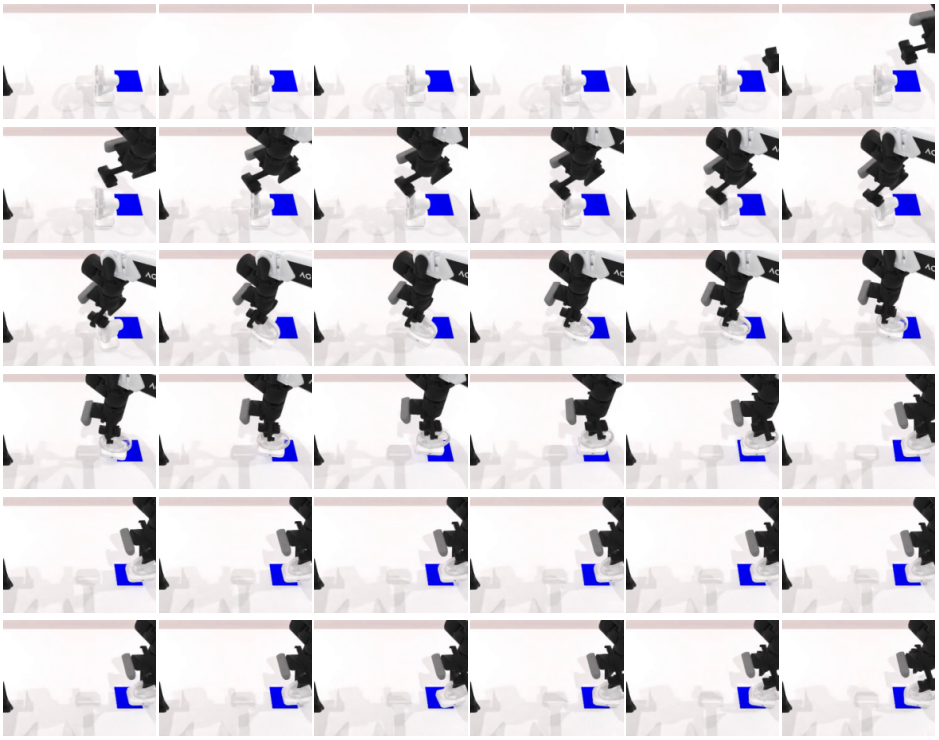

Figure 16: Illustration of `place_fan` on RoboTwin

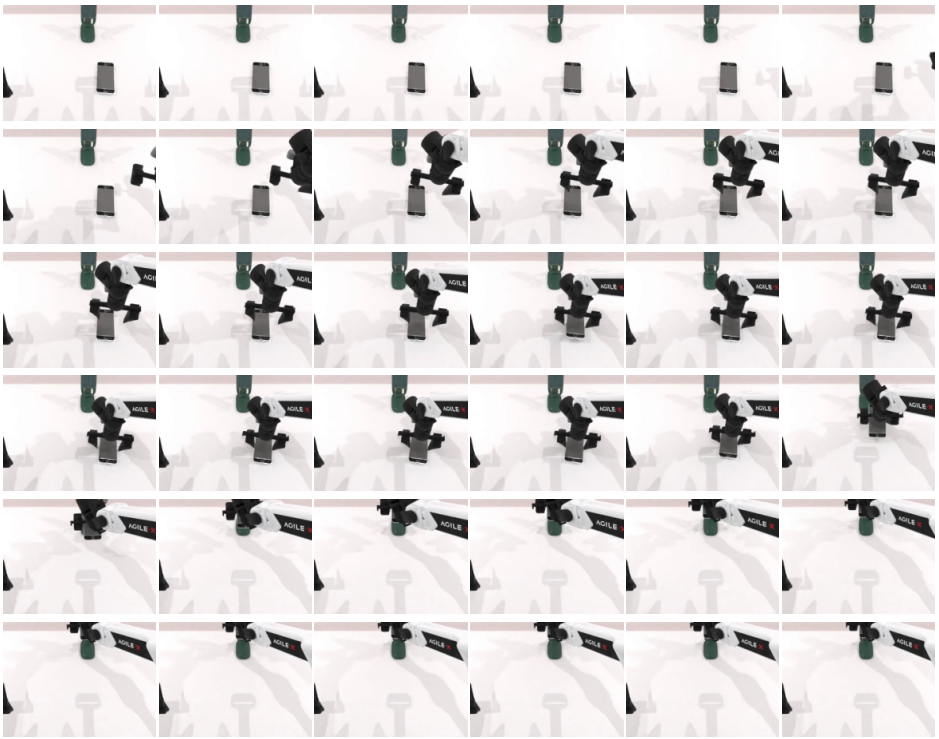

Figure 17: Illustration of `place_phone_stand` on RoboTwin

Figure 18: Illustration of `put_object_cabinet` on RoboTwin

Figure 19: Illustration of `click_alarmclock` on RoboTwin

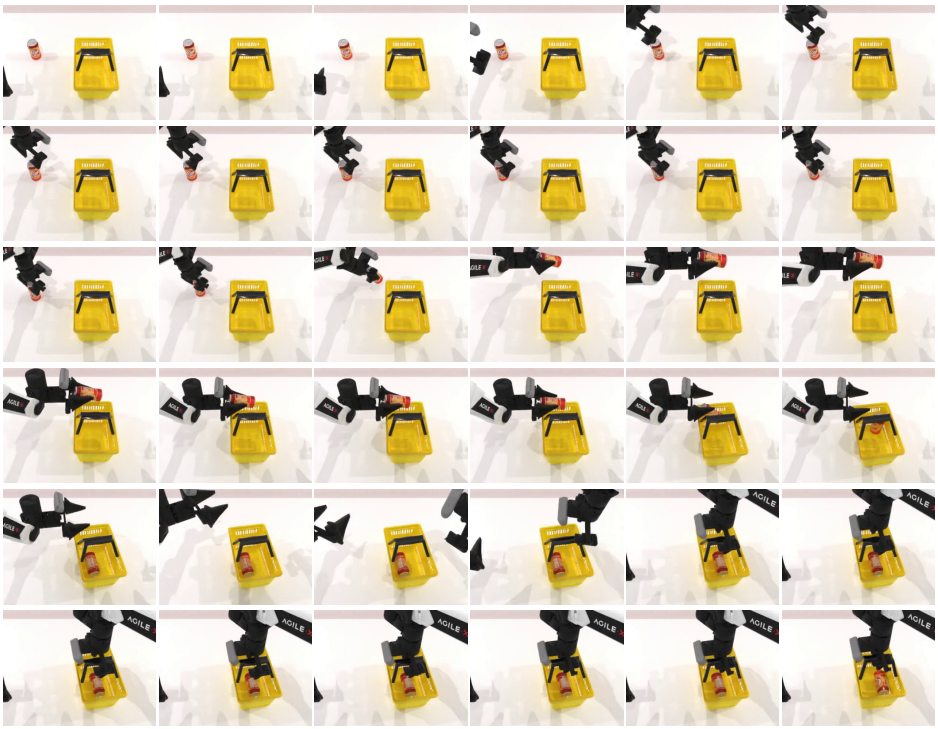

Figure 20: Illustration of `place_can_basket` on RoboTwin

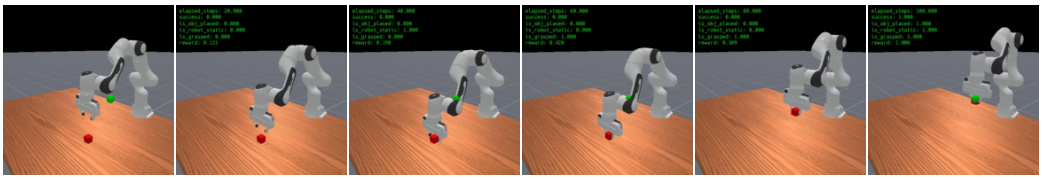

Figure 21: Illustration of `PickCube-v1` on ManiSkill

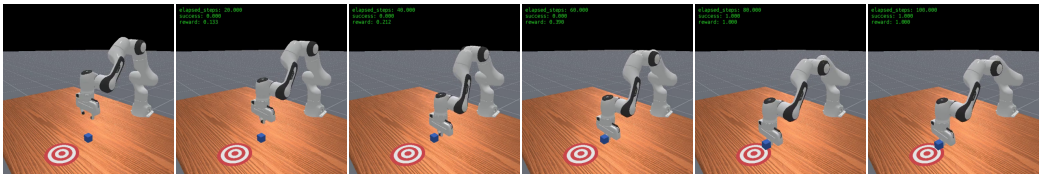

Figure 22: Illustration of `PushCube-v1` on ManiSkill

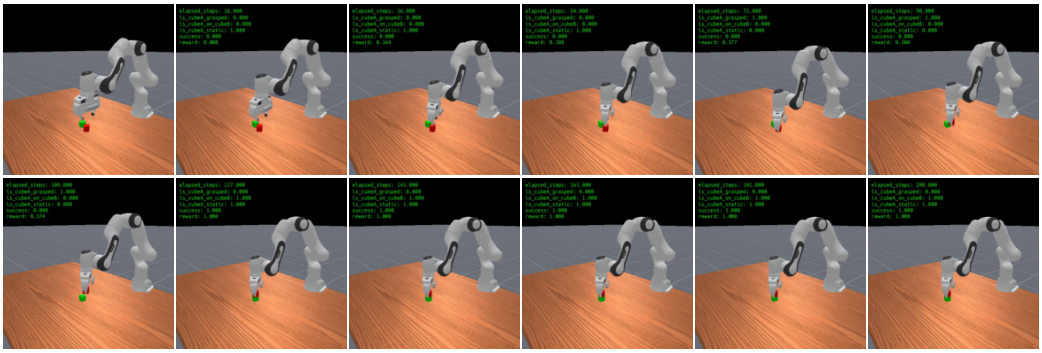

Figure 23: Illustration of `StackCube-v1` on ManiSkill

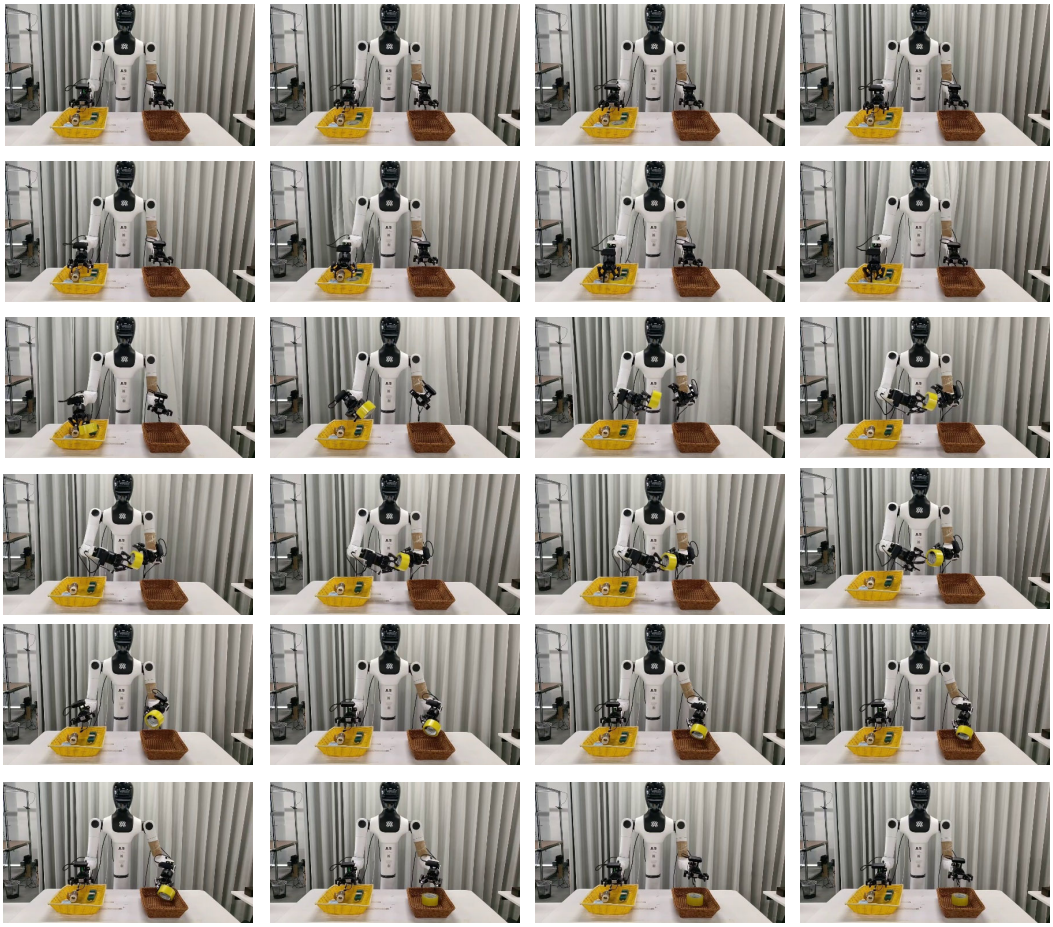

Figure 24: Illustration of real-world task: `find_out_packaging_tape_into_the_other_basket`.

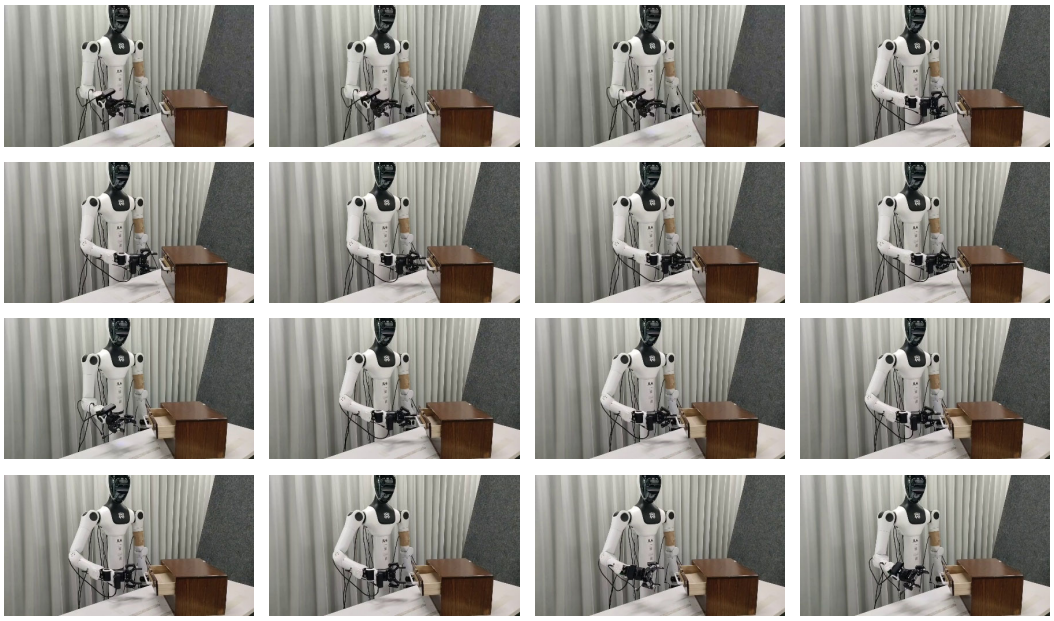

Figure 25: Illustration of real-world task: `open_white_solid_wood_drawers`.

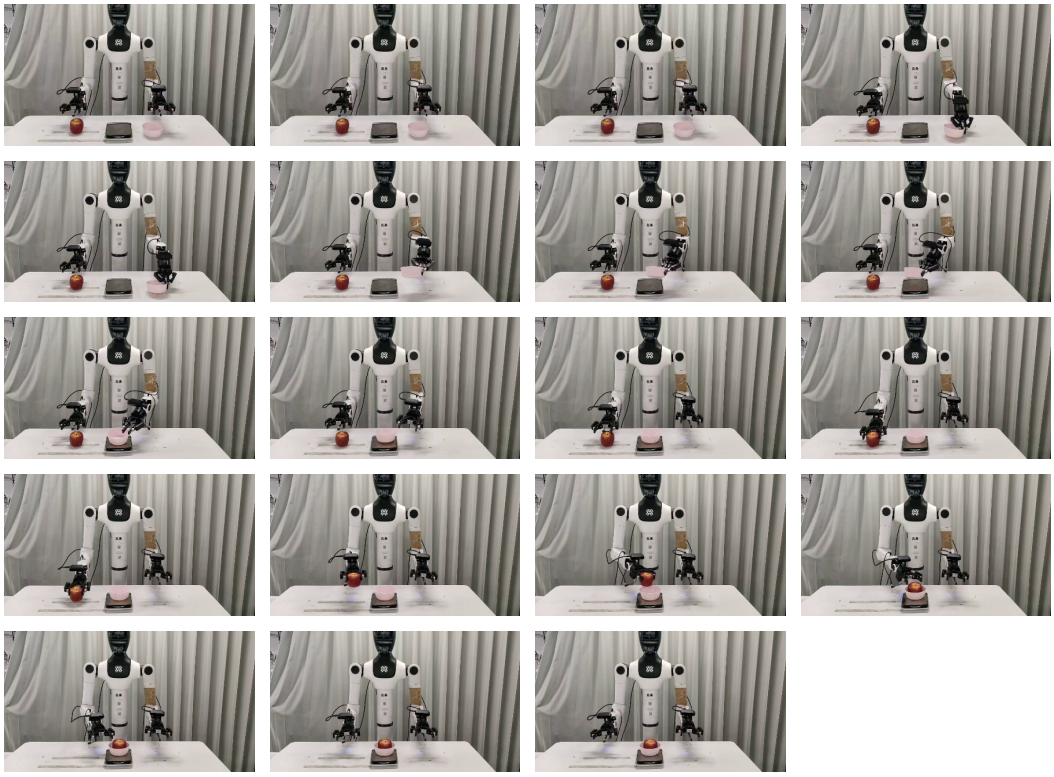

Figure 26: Illustration of real-world task: `pick_up_weigh`.

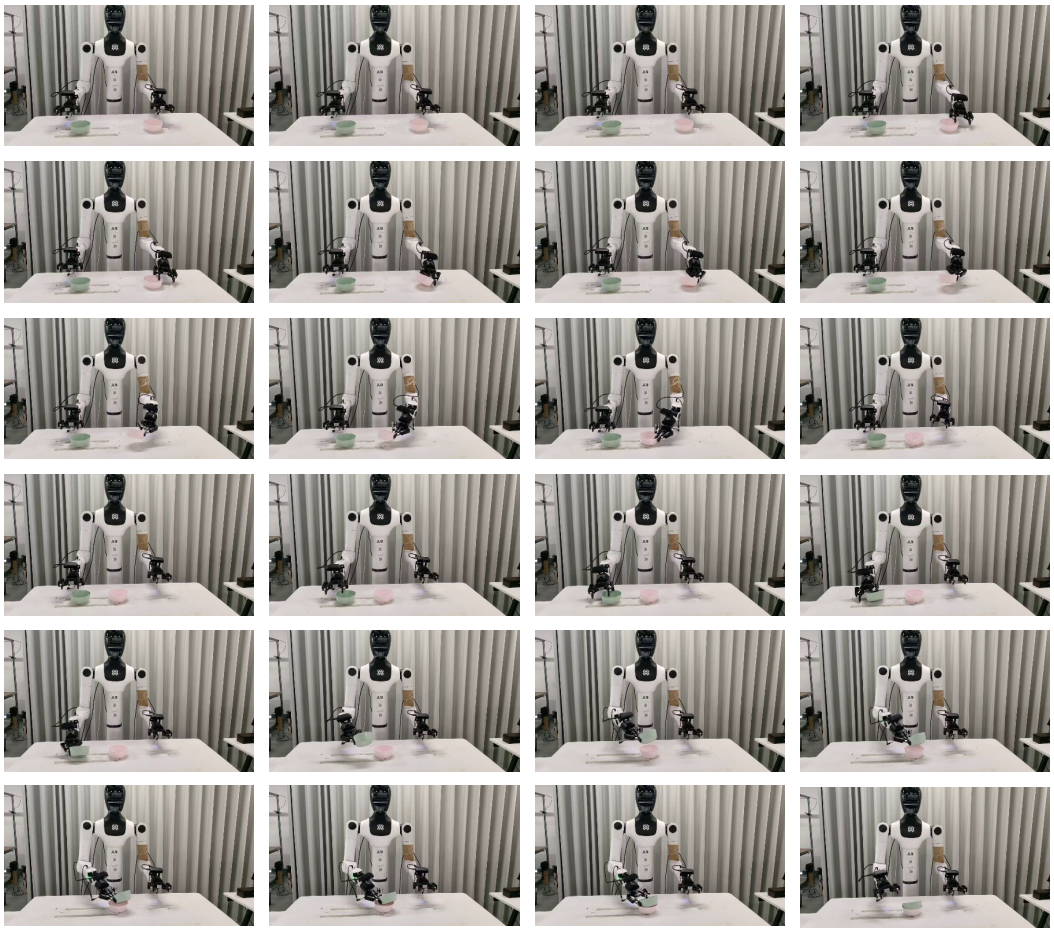

Figure 27: Illustration of real-world task: `stack_bowl`.

H Detailed Literature Review

The landscape of robot learning has undergone a profound transformation in recent years, driven by the shift from classical modular pipelines to end-to-end sensorimotor policies. Central to this evolution is the integration of high-capacity architectures and generative modeling techniques, which have demonstrated unprecedented capabilities in continuous control, spatial reasoning, and open-world generalization. To systematically navigate these advancements, this section categorizes the recent literature into two primary paradigms based on their architectural and semantic scope: Vision-Action (VA) models, which focus on direct, high-fidelity sensorimotor mappings, and Vision-Language-Action (VLA) models, which leverage large-scale semantic foundations to achieve cognitive, generalized control across diverse environments and embodiments.

H.1 Vision-Action Models (VA)

Visual-motor control relies heavily on learning robust mappings directly from high-dimensional visual observations to continuous action spaces. In recent years, both stochastic diffusion processes and deterministic sequence modeling have driven significant progress in vision-action formulations, proving that highly capable control primitives can be synthesized without explicit linguistic grounding.

Compact architectures employing convolutional or transformer encoders coupled with diffusion heads have established a new standard for action generation. Foundational frameworks like Diffusion Policy [1] demonstrated the core efficacy of this approach, which was rapidly expanded by DP3 [46] to handle 3D point-cloud inputs for superior spatial generalization, and subsequently by iDP3 [47] to enable humanoid robots to learn from noisy human demonstrations. The architectural efficiency and representational power of these systems are continually refined. Models such as Mamba Policy [48] introduce linear-complexity backbones, while H^3DP [29] explicitly incorporates hierarchical structures to bind visual features with action generation. Modality integration is further enhanced by MCDP [49] and GCP [50], which leverages compositional diffusion for superior performance. To handle specialized and spatially complex tasks, DexDiffuser [51] employs progressive denoising for multi-fingered grasp synthesis, 3D Diffuser Actor [52] conditions control on tokenized 3D scene representations, and R&D [53] presents a unified image-action formulation using ViT encoders.

Overcoming the challenges of long-horizon and structured planning requires dedicated temporal and kinematic strategies. ALOHA Unleashed [54] trains transformers with a diffusion loss for complex bimanual skills, ChainedDiffuser [30] connects predicted end-effector keyposes with feasible trajectories, and HDP [28] injects kinematics-aware priors to ensure physical realism. Furthermore, models like S^2-Diffusion [55] and C3DM [56] utilize visual foundation priors and constrained-context conditioning to resist environmental distractors. To optimize training and inference, Streaming Diffusion Policy [57] and Bidirectional Decoding (BID) [58] accelerate policy synthesis and enable test-time closed-loop adaptation, while Crossway Diffusion [59] and Equivariant Diffusion Policy [60] exploit domain symmetries and auxiliary objectives for enhanced sample efficiency.

Beyond iterative denoising, autoregressive sequence modeling has emerged as a dominant non-diffusion strategy to mitigate compounding errors. Frameworks like ACT [2] predict entire action sequences, while Chunking Causal Transformer [61] introduces trajectory segmentation to improve stability for long-horizon execution. Spatial grounding remains a critical axis of improvement, with Act3D [62] and Spatial Policy [63] explicitly modeling scene geometry to align visual predictions with executable motions. Recently, flow matching has surfaced as a highly efficient alternative to diffusion. Frameworks including ManiFlow [64], Flow Matching Policy Gradients [65], and VITA [66] enable dexterous action synthesis in minimal steps and seamlessly integrate with reinforcement learning. Finally, physical consistency and safety in VA models are bounded by mechanisms like DynaGuide [67], which incorporates external dynamics feedback, and Latent Policy Barrier (LPB) [68], which formulates implicit safety boundaries to distinguish in-distribution states from out-of-distribution risks.

H.2 Vision-Language-Action Models (VLA)

As robotic systems transition toward open-world generalists, integrating semantic reasoning with low-level control has catalyzed the development of Vision-Language-Action architectures. By leveraging large-scale foundation models and broad cross-embodiment corpora, these systems marry deep semantic understanding with probabilistic or autoregressive action generation. Language models frequently act as high-level cognitive engines that synthesize plans and decompose tasks, delegating precise motor execution to specialized action heads.

The integration of pretrained semantic knowledge fundamentally alters how policies are structured. Models like MDT [69], StructDiffusion [70], and PlayFusion [71] fuse object-centric transformers with downstream execution, while systems like ROSIE [72] exploit text-to-image diffusion for aggressive, targeted data augmentation. To tighten the loop between high-level reasoning and low-level control, compositional planning stacks such as HiP [73] and Plan Diffuser [74] compose expert LLMs for task planning with video diffusion for trajectory proposals, autoregressively emitting textual subgoals that translate into visual targets. To maximize efficiency, TinyVLA [75] freezes a multimodal backbone and applies parameter-efficient tuning to produce accurate actions with minimal computational overhead.

The pursuit of cross-embodiment generalization has spurred massive pretraining initiatives on heterogeneous robot datasets. Frameworks like Octo [42], Diffusion-VLA [76], and LAPA [77] aggregate massive datasets from the Open X-Embodiment repository, training transformer backbones to map multimodal instructions directly to action tokens across drastically different hardware platforms. Data scaling is equally pivotal in non-diffusion generalists, where the lineage of RT-1 [18], RT-2 [78], and RT-X [79] demonstrates that transferring web-derived vision-language knowledge into control establishes state-of-the-art performance bars. Models like ChatVLA [80] and RDT 1&2 [39, 81] advance this paradigm by co-training on both robotic execution and abstract reasoning data, utilizing mixture-of-experts routing to prevent task interference while unifying action spaces across heterogeneous embodiments.

Modern VLA architectures are increasingly blurring the lines between discrete reasoning and continuous control. The π_0 [3], $\pi_{0.5}$ [21], and $\pi_{0.7}$ [22] frameworks couple pretrained vision-language backbones with flow-matching action experts for precise manipulation and broad generalization. HybridVLA [82] unifies the continuous nature of diffusion with the contextual reasoning of autoregression within a single large language model. Spatial and relational reasoning are elevated by systems like 3D-VLA [83], **LEO**, and SpatialBot [84], which ground multimodal cognition in 3D environments. Furthermore, dual-system approaches like GR00T N1 [85], Galaxea G0 [86], and RoboDual [32] separate yet synergize multimodal planning and low-level execution for humanoid robots. Finally, constraint-driven representations such as ReKep [87] and VoxPoser [88] leverage large language models to extract affordances and spatial rules from natural language, dynamically composing 3D value maps that guide robotic interactions. Ultimately, these advanced VLAs converge on a unified recipe where language models structure objectives, generative processes synthesize trajectories, and massive pretraining supplies the indispensable semantic and physical priors required for true open-world adaptability.

H.3 World-Action Models (WAM)

The pursuit of truly autonomous agents has catalyzed the emergence of World-Action Models (WAM), a paradigm that extends beyond reactive control by explicitly internalizing the physical dynamics of the environment. Pioneered by frameworks like UniPi [89], which frames sequential decision-making as a text-conditioned video generation problem, this lineage posits that predicting future visual states is fundamentally intertwined with robust action synthesis. By explicitly forecasting the consequences of interactions, these models construct a predictive understanding of the world that serves as a powerful prior for grounded manipulation and navigation.

Building on this generative foundation, early approaches decouple or tightly couple the prediction and execution phases. Methods such as Gen2Act [90] leverage zero-shot human video generation

to synthesize a visual plan, which subsequently guides a conditioned policy during execution. Conversely, unified architectures optimize for both modalities in an end-to-end manner. GR-1 [91] introduces a GPT-style formulation that takes language instructions, historical observations, and robot states to simultaneously predict continuous motor commands and future image sequences. This dual-objective learning is further advanced by WorldVLA [92] and RynnVLA-002 [93], which explicitly leverage the synergy between action and image understanding to capture underlying environmental physics, thereby significantly improving action generation. Similarly, Lingbot-VA [94] employs an autoregressive diffusion framework to concurrently learn frame prediction and policy execution.

Recognizing the immense wealth of physics priors embedded in broad visual data, recent efforts increasingly exploit large-scale video pretraining. GR-2 [95] demonstrates that pretraining on vast corpora of Internet videos allows the model to capture universal dynamics before specializing in robotic tasks. Cosmos Policy [96] elegantly adapts a massive pretrained video foundation model directly into an effective robot policy through a single post-training stage, requiring no architectural modifications. Along a similar trajectory, DreamGen [97] repurposes advanced image-to-video generative models to synthesize photorealistic task executions across novel environments, while Genie Envisioner [98] utilizes a large-scale, instruction-conditioned video diffusion space to capture the spatial, temporal, and semantic dynamics of real-world interactions. Pushing this integration further, DreamZero [99] builds its entire WAM architecture directly upon a pretrained video diffusion backbone, maximizing the transfer of visual-temporal knowledge to the control domain. DreamDojo [100] chooses another way that learns diverse interactive dexterous controls from egocentric human videos. GigaWorld [101] proposes an action-centered training paradigm without overreliance on explicit video synthesis.

However, integrating high-dimensional video forecasting with low-dimensional action spaces introduces significant modality conflicts, prompting sophisticated architectural innovations. DUST [102] resolves this through a dual-stream diffusion mechanism, handling the inherent divergence between modalities and elevating performance across diverse tasks. Mixture-of-Experts designs have also proven highly effective; Motus [103] and MotuBrain [104] leverage three-stream Mixture-of-Transformers architectures combined with UniDiffuser-style scheduling to orchestrate flexible, joint modeling of visual understanding, video generation, and action synthesis. To manage the computational overhead of these complex systems, Fast-WAM [20] retains the representational benefits of video co-training but strategically bypasses future prediction during test-time inference. Meanwhile, RoboDreamer [105] tackles complexity by factorizing video generation to learn a highly compositional world model, and Flare [106] enables diffusion transformers to anticipate latent representations of future observations, facilitating nuanced, long-term temporal reasoning without the burden of pixel-perfect rendering.

The culmination of these advancements is realized in large-scale, unified frameworks that establish a closed perception-prediction-action loop. Models like Vidar [107] and DreamVLA [108] pair embodied video diffusion priors with inverse dynamics models, seamlessly translating forecasted world knowledge into executable motor commands. Act2Goal [109] similarly integrates a goal-conditioned visual world model with multi-scale temporal control for robust, generalized manipulation. At the frontier of architectural scaling, LDA-1B [110] and Unified World Models (UWM) [111] demonstrate that universal embodied data ingestion—jointly optimizing dynamics, policy, and visual forecasting—yields immensely capable robot foundation models. Alongside them, CoT-VLA [112] scales to billions of parameters, fluently generating intertwined visual and action tokens to master complex embodied reasoning and secure state-of-the-art performance across dynamic environments.