

Vision language action models for robotic manipulation: A comparative evaluation of policy and value architectures

Hongliang Wang¹, Jin Liu³, Xiaolin Zhu⁴, Qiang Fu¹, Yuhua Zheng¹, Wangmeng Zuo⁵, Changqing Zou⁶, Evgeny Burnaev²

¹ Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences, Hangzhou 310024, China, ³ Hangzhou Dianzi University, Hangzhou 310018, China, ⁴ College of Electrical and Information Engineering, Hunan Institute of Engineering, Xiangtan 411104, China, ¹ Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences, Hangzhou 310024, China, ¹ Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences, Hangzhou 310024, China, ⁵ School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China, ⁶ Zhejiang Lab, Hangzhou 311121, China, ² Artificial Intelligence Center, Skolkovo Institute of Science and Technology, Moscow 121205, Russia

Vision Language Action (VLA) models have become an important approach to robotic manipulation by integrating visual observations, language instructions, and robot states for action generation. Nevertheless, behavior cloning-based VLA policies remain vulnerable to distribution shifts, accumulated execution errors, and failure recovery difficulties during long horizon closed-loop operation. This paper investigates whether an auxiliary value architecture can improve the reliability of VLA policy learning without replacing the policy as the low-level controller. We present a controlled real-robot study on a LeRobot SO-101 platform covering block grasping, placing, and sorting tasks. The study compares three policy architectures, namely ACT, diffusion policy, and SmolVLA, and two value architectures, namely ValueLite and ValueVision. The value models estimate state quality, identify early failure precursors, and generate Advantage-Conditioned Policy (ACP) labels for policy retraining. Our evaluation considers policy-only success, human-assisted success, intervention rate, recovery success, completion time, action smoothness, value prediction error, early failure detection, and inference cost. In addition, data-controlled and mechanism-level ablations are designed to separate the effects of additional recovery or rollout data from the effects of learned value signals and ACP conditioning. The results show that value models provide useful trajectory ranking and failure-detection signals in the studied manipulation setting. Value-guided variants improve assisted task completion and reduce takeover frequency relative to recovery-only training, although these gains must be distinguished from fully autonomous policy performance. Overall, the study supports value-guided VLA retraining as a practical approach for improving closed-loop manipulation reliability, while also identifying the limitations imposed by task diversity, evaluation scale, model size, and the reliability of the value target.

Keywords: Advantage-conditioned policy, closed-loop evaluation, robotic manipulation, value architecture, vision language action models

1. INTRODUCTION

Robotic manipulation requires an agent to interpret task instructions, perceive object and robot states, and generate temporally-consistent control actions under physical uncertainty. These requirements become more difficult in long horizon tasks, where small action deviations may accumulate and eventually cause grasping failures, object displacement, or loss of task progress. Traditional imitation-learning policies, including Action Chunking with Transformers (ACT) and diffusion policy, learn mappings from multimodal observations to action sequences through behavior cloning [1]. ACT predicts temporally-correlated action chunks [2], whereas diffusion policy represents multimodal action distributions through conditional diffusion [3]. These approaches can perform effectively in single-task or relatively structured environments, but their performance may degrade under language variation, task switching, execution perturbations, and state distributions that differ from the demonstrations.

Vision language action (VLA) models extend this paradigm by combining visual lan-

guage representations with robot action generation. Representative systems, including SmolVLA, OpenVLA, Octo, GR00T, and the π_0 family, use different forms of pretrained visual, language, and action representations to improve task conditioning and cross-scene transfer [4, 5, 6, 7, 8]. However, these models do not eliminate the difficulties associated with real-robot deployment. In particular, a VLA policy may experience inference latency, compounding errors, incomplete coverage of failure states, and distribution shifts caused by imperfect demonstrations or unsuccessful rollouts. These problems are especially apparent in long horizon closed-loop manipulation, where a policy must continuously recover from small deviations rather than simply produce a correct action at a single time step.

Most policy-learning pipelines treat the control loop primarily as an action-generation problem. Once the policy has been trained, its predictions are executed sequentially, while the quality of the current state and the likelihood of future failure are not explicitly modeled. This separation between action generation and state evaluation limits the ability of the learning system to identify low-quality trajectory segments, preserve useful recovery behavior, and use failure information during policy retraining.

This paper studies a complementary policy–value formulation for real-robot VLA manipulation. Instead of using a value model as an additional low-level controller, we use it as an auxiliary evaluator of visual-proprioceptive states and trajectory segments. The value model estimates a task-progress and failure-risk surrogate, which can be used to rank states, identify early failure precursors, and construct training labels for policy improvement. In particular, we convert value-based advantage estimates into discrete Advantage-Conditioned Policy (ACP) labels. These labels are used to distinguish higher-quality and lower-quality trajectory segments during VLA policy retraining. This formulation is related to value-guided imitation learning and offline reinforcement learning [9, 10, 11, 12, 13], but the present study focuses on its implementation and evaluation in a real-robot VLA pipeline.

We construct a controlled experimental framework with five functional stages: data collection, value target construction, value model training, ACP annotation, and policy retraining. The framework is evaluated on a physical LeRobot SO-101 platform using block manipulation tasks, including grasping, placing, and sorting. The policy comparison includes ACT, diffusion policy, and SmolVLA. The value comparison includes ValueLite, a lightweight visual-proprioceptive value model, and ValueVision, a vision-enhanced value model. To isolate the contribution of value guidance, we use data-controlled comparisons that fix the training data pool while varying the use of learned value signals and ACP conditioning. Additional control groups quantify the effects of adding recovery and rollout data. In addition to aggregate success rate, we report policy-only success, human-assisted

success, intervention rate, recovery success, completion time, action variation, value prediction error, early failure detection, and inference cost.

The scope of this work is deliberately limited. The experiments are conducted on one SO-101 platform and primarily evaluate structured block manipulation tasks. Therefore, the paper does not claim that the investigated policies or value models provide universal performance across robot embodiments, open-world tasks, or all current VLA architectures. Recent generalist policies, including the π_0 family, are considered in the broader architectural context and will be included in the quantitative comparison only when a compatible implementation and a fair evaluation protocol are available. The main objective is instead to determine whether value-guided policy retraining provides a measurable and reproducible benefit under a shared real-robot evaluation setting.

This study addresses the following research questions:

1. Under a shared observation interface, action space, demonstration split, and evaluation protocol, how do ACT, diffusion policy, and SmolVLA differ in closed-loop success, action stability, completion time, and inference cost?
2. How effectively can ValueLite and ValueVision estimate state quality, separate successful and failed trajectory segments, and identify failure precursors before task failure or human intervention?
3. After controlling for the training data pool, does ACP conditioning generated by a learned value model provide an incremental benefit beyond the gains obtained from adding recovery and rollout data?
4. What are the practical trade-offs between lightweight and vision-enhanced value models in terms of prediction quality, intervention reduction, autonomous performance, latency, memory usage, and deployment cost?

The contributions of this work are as follows:

1. We establish a controlled real-robot evaluation protocol for comparing policy and value components in VLA-based block manipulation. The protocol uses shared observation, action, task, and evaluation interfaces and separates fully autonomous performance from human-assisted completion.
2. We compare a lightweight value model and a vision-enhanced value model using multiple criteria, including value prediction error, success–failure separation, early failure area under the ROC curve, lead time, ACP precision, inference latency, and computational cost.
3. We formulate a data-controlled evaluation of ACP-based policy retraining. The protocol distinguishes the effect of additional recovery or rollout data from the incremental effect of learned value labels and includes ablations for random labels, value-guided labels, filtering-only, and prompt-only mechanisms.
4. We identify the conditions and limitations under which value-guided VLA retraining can improve closed-loop

manipulation reliability. In particular, we distinguish assisted task completion and intervention reduction from fully autonomous policy-only success, and we report uncertainty and statistical evidence rather than relying solely on aggregate success-rate differences.

2. RELATED WORK

2.1 Imitation learning for robot manipulation

Behavior Cloning (BC) learns a mapping from robot observations to demonstrated actions by minimizing an action prediction loss [1]. It has been widely used in robot manipulation because it can learn task-specific behaviors from relatively small demonstration datasets. However, one-step action prediction can accumulate small errors during long horizon execution. Once the robot state deviates from the demonstration distribution, the policy may produce actions for states that were rarely observed during training. This distribution-shift problem can lead to trajectory drift, unstable contact, and failure to recover after an execution error.

Action Chunking with Transformers (ACT) addresses part of this problem by predicting a temporally-ordered block of future actions instead of an isolated action at each control step [2]. The action chunking formulation can improve short-term temporal consistency and reduce the sensitivity to individual prediction errors. Nevertheless, action chunking alone does not explicitly evaluate whether the current state is progressing toward task completion. Its performance may therefore still degrade when the robot encounters unfamiliar object configurations, contact states, or recovery situations.

A diffusion policy formulates action generation as conditional de-noising over a sequence of continuous actions [3]. By modeling a conditional action distribution, it can represent multiple feasible motion patterns and avoid the averaging behavior that may occur with deterministic regression. This property is useful for manipulation tasks in which the same object can be approached through different trajectories. However, diffusion-based action generation introduces additional inference steps and may be sensitive to control frequency, inference latency, and closed-loop state deviations. In this paper, ACT and diffusion policy are used as action-generation baselines, while the main VLA policy is evaluated separately.

2.2 Offline reinforcement learning and value-guided imitation

The use of value information to improve imitation learning is closely related to offline reinforcement learning and advantage-weighted policy optimization. Advantage-Weighted Regression (AWR) re-weights behavioral data according to estimated advantages, thereby assigning greater influence to actions associated with higher returns [9]. Advantage-Weighted Actor-Critic (AWAC) extends this idea to combine offline data with subsequent online interaction [10]. Q-filtered behavior cloning uses

a learned action-value function to retain or emphasize actions that are estimated to be better than the current policy [11]. Implicit Q-Learning (IQL) avoids direct maximization over out-of-distribution actions and uses expectile regression to extract value information from offline datasets [12]. A decision transformer provides another formulation by conditioning action prediction on desired returns rather than explicitly learning a conventional value function [13].

These methods provide important foundations for using trajectory quality or advantage estimates during policy learning. The present study is related to this line of work but has a different experimental focus. We do not propose a new general offline reinforcement learning algorithm, nor do we directly optimize a reinforcement learning objective over an unconstrained action space. Instead, an auxiliary value model receives visual observations, robot states, and task information, estimates the quality of the current state, and produces discrete Advantage-Conditioned Policy (ACP) labels. These labels are used for trajectory filtering and policy conditioning while the underlying VLA policy continues to be trained with its original imitation-learning objective.

This distinction is important for real-robot experiments. The value model is treated as an auxiliary signal generator rather than as a replacement for the policy controller. Consequently, the contribution of the value model must be separated from the effects of adding recovery or rollout data. The revised experiments therefore include data-matched and label-matched ablations to evaluate whether improvements arise from additional data, value-based sample selection, ACP conditioning, or their combination.

2.3 Intervention learning and recovery policies

A major source of difficulty in real-robot learning is that failures often occur outside the distribution covered by successful demonstrations. Dataset Aggregation (Dagger) addresses this problem by iteratively collecting expert labels on states visited by the learned policy. Related intervention-based approaches allow a human operator to take control when the policy enters an unsafe or unrecoverable state, and then use the resulting corrective behavior for subsequent policy updates. These approaches have been applied to autonomous driving, manipulation, and other sequential decision-making problems.

Recovery learning focuses more specifically on generating actions that return the system from a deviated state to a task-progressing state. Recovery demonstrations can contain information that is absent from ordinary successful demonstrations, including contact correction, re-approach behavior, object repositioning, and post-failure stabilization. Such data can improve the coverage of the training distribution, but the benefit may be caused by the additional data itself rather than by any value-guided mechanism.

The present work uses recovery trajectories as an explicit control condition. SmolVLA-Recovery is trained with additional recovery data but without value-derived ACP labels. Value-guided variants use the same controlled data pool and add value-based filtering or ACP conditioning. This design separates recovery-data expansion from the use of an auxiliary value signal. Unlike standard Dagger, the current study does not assume that every visited state is relabeled by an expert. Instead, it focuses on how existing intervention and recovery trajectories can be evaluated and reused during VLA retraining.

2.4 Vision language action policies and robot foundation models

Vision language action policies extend robot learning by conditioning action generation on visual observations, language instructions, and robot state. Earlier robot foundation models such as RT-1 and RT-2 demonstrated the potential of learning language-conditioned manipulation policies from large-scale robot and vision language data. PaLM-E incorporated visual and sensor observations into a large language-model-based embodied reasoning system, while SayCan combined language-model reasoning with affordance or value estimates to select feasible robot skills.

More recent generalist policies, including Octo, OpenVLA, π_0 , and GR00T, investigate large-scale pretraining, cross-task transfer, cross-embodiment learning, and language-conditioned action generation. These models differ substantially in backbone, model size, pretraining data, robot embodiment, action representation, and fine-tuning protocol. Therefore, numerical comparisons across these systems should be interpreted carefully unless the observation interface, task data, hardware platform, and evaluation protocol are controlled.

SmolVLA is used in this paper as the primary VLA policy because it provides a practical vision language action interface and can be trained within the experimental data and hardware constraints of the SO-101 platform. The role of SmolVLA in this study is not to represent all VLA models. Instead, it provides a common policy backbone for evaluating whether an auxiliary value architecture can improve closed-loop manipulation under a fixed task and robot setting. ACT and diffusion policy are included as non-VLA action-generation baselines, allowing the experiments to compare action chunking, conditional diffusion, and language-conditioned policy learning under a shared observation and action interface.

2.5 Value models for robotic manipulation

Value-related models have been used in robotics for several purposes, including success prediction, reward estimation, affordance prediction, trajectory ranking, failure detection, and intervention timing. A success classifier can estimate whether an episode or state is likely to lead

to task completion. A reward or value model can provide denser supervision than a terminal success label. Visual affordance models can estimate whether an object or region is suitable for a particular manipulation action. Failure-prediction models can identify abnormal visual or proprioceptive patterns before a task failure becomes irreversible.

However, a value model for real-robot manipulation must address several sources of uncertainty. Contact dynamics, occlusion, lighting changes, object displacement, and sensor noise can make the same nominal state appear different across trials. In addition, a state may be recoverable even if it differs substantially from a successful demonstration distribution. For this reason, the value model should not be evaluated only by scalar prediction error. Ranking quality, success-failure separation, early failure detection, calibration, lead time, and downstream policy utility are also relevant.

In this paper, ValueLite and ValueVision are introduced as two value-model variants within the proposed evaluation framework. ValueLite combines visual features, robot state features, and task information using a lightweight visual encoder and a distributional value head. It is intended to provide a low-cost value estimate for frequent training iterations and online diagnosis. ValueVision uses two complementary visual encoders, DINOv2 and SigLIP, to examine whether geometric and semantic visual features improve state quality estimation in manipulation scenes involving object relations, occlusion, and language-conditioned targets.

Both models share the same value-target definition, output interface, and evaluation protocol. Their architectural difference is therefore used to study the trade-off between visual representation capacity, value prediction quality, and computational cost. The detailed feature-fusion equations, value-target construction, distributional learning objective, and ACP generation procedure are presented in Section 3.

2.6 Positioning of this work

This paper studies the interaction between policy architecture, value architecture, and recovery-oriented data in real-robot VLA manipulation. Its scope is narrower than a general comparison of all robot foundation models. The contribution is organized around three controlled questions.

First, under a shared observation interface, action space, demonstration dataset, and evaluation protocol, how do ACT, diffusion policy, and SmolVLA differ in closed-loop manipulation behavior? Second, when the value target and training data are fixed, how do a lightweight value model and a vision-enhanced value model differ in state quality estimation and early failure detection? Third, after controlling the policy-retraining data pool, does value-derived ACP provide an additional benefit beyond the benefit of adding recovery or rollout data?

The work is therefore complementary to offline reinforcement learning, intervention learning, and robot foundation-model research. It does not claim that ACP replaces advantage-weighted policy optimization, that the proposed value target is a universally valid return function, or that the evaluated policy architectures are representative of all current VLA systems. Instead, it provides a controlled case study of how an auxiliary value model can be integrated into VLA retraining for long horizon block manipulation on a real robotic platform.

3. METHOD

3.1 Overall framework

This paper studies whether an auxiliary value model can improve the closed-loop reliability of a VLA policy during real-robot manipulation. The value model is not used as a low-level controller. Instead, it evaluates the quality of observed states and generates auxiliary signals for trajectory selection and policy retraining.

The framework contains four stages. First, policy models are trained using the standardized observation and action interfaces described below. The policy comparison includes ACT, diffusion policy, and SmoVLA. Second, value models are trained from trajectory-level and frame-level supervision obtained from successful, failed, and recovery episodes. The value-model comparison includes ValueLite and ValueVision. Third, the trained value model estimates the quality of states in the policy-retraining data and generates Advantage-Conditioned Policy (ACP) labels. Finally, SmoVLA is retrained using the controlled data pool with or without ACP signals.

The experimental design separates three factors that may otherwise be confounded:

1. the policy architecture;
2. the amount and composition of policy-retraining data;
3. the use of learned value signals and ACP conditioning.

Accordingly, the recovery-only baseline is compared with a data-expanded behavioral-cloning baseline before evaluating value-guided variants. This design allows the contribution of the value signal to be evaluated beyond the effect of adding recovery or rollout data.

The complete data flow is summarized as follows:

1. Collect successful demonstrations, closed-loop rollouts, and recovery trajectories;
2. train the policy and value models using episode-level data partitions;
3. estimate state quality values for training frames;
4. compute advantage scores and construct ACP labels;
5. retrain SmoVLA using identical data and optimization settings across controlled variants;
6. evaluate autonomous execution, human-assisted execution, intervention frequency, and recovery behavior on an independent evaluation set.

3.2 Multimodal observation interface

To make the policy and value-model comparisons as consistent as possible, we define a shared multimodal observation interface $\mathcal{O}$. Each observation consists of two RGB images, robot proprioceptive states, and a natural-language task description:

$$o_t = (I_t^g, I_t^w, q_t, g_t, c), \quad (1)$$

where I_t^g denotes the global-view image, I_t^w denotes the wrist-camera image, q_t denotes the robot joint configuration, g_t denotes the gripper state, and c denotes the task description.

The global camera provides information about the object layout, tray position, workspace context, and target location. The wrist camera provides local information about object geometry, contact configuration, gripper alignment, and partial occlusion near the end effector.

The visual inputs are synchronized at each control step and processed using the preprocessing procedure specified in the experimental configuration. Unless otherwise stated, the images are resized to 224×224 pixels. The proprioceptive state contains six dimensions: five joint-position values and one continuous gripper-opening value. The same state definition and normalization procedure are used for all policy and value models.

The proprioceptive input is read directly from the robot bus. No additional velocity estimation or temporal filtering is applied in the common interface, so that the comparison does not introduce different temporal preprocessing pipelines across models. Model-specific visual encoders may apply their own standard feature extraction after the shared input preprocessing.

3.3 Policy architectures

We compare three policy architectures under the common observation interface and action space.

3.3.1 ACT

ACT predicts a temporally-ordered action chunk rather than an isolated action at every control cycle. This formulation can improve short-term temporal consistency and reduce the sensitivity to individual action prediction errors. In the present study, ACT is trained using the demonstration dataset and evaluated as a non-VLA imitation-learning baseline.

3.3.2 Diffusion policy

Diffusion Policy generates an action sequence through conditional de-noising. Its conditional action distribution can represent multiple feasible trajectories for the same task and observation. The method is evaluated using the same visual inputs, robot state definition, task descriptions, and action representation as the other policy baselines. The additional inference cost associated with iterative de-noising is reported separately.

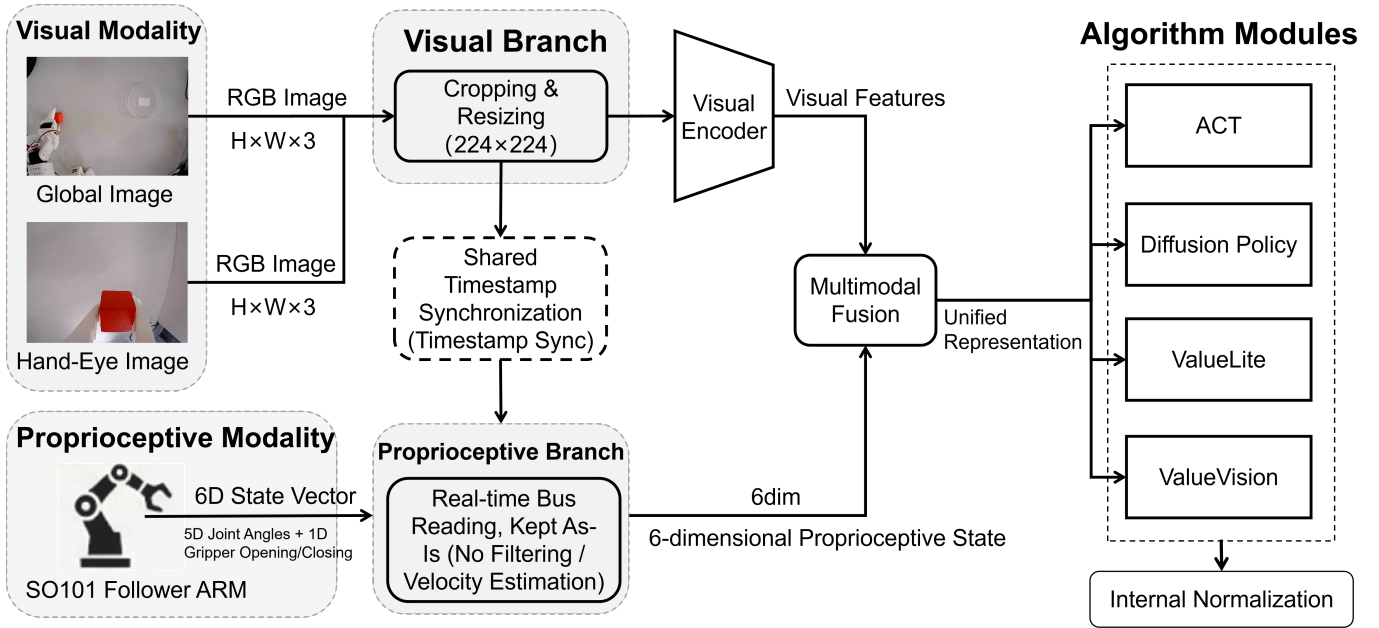


Figure 1 – The modular pipeline of the multimodal observation interface. The visual modality encompasses asymmetric dual-camera views cropped to standard shapes, while the proprioceptive branch continuously ingests raw bus vectors from the hardware follower arm.

3.3.3 Smo/VLA

SmoVLA receives visual observations, robot states, and natural language task descriptions, and generates continuous robot actions or action sequences. It is used as the primary VLA policy because it provides a common interface for behavior-cloning, recovery-data retraining, and ACP conditioning.

The policy comparison is intended to evaluate behavior under a shared local robot-learning setup. It is not interpreted as a ranking of all existing VLA or robot foundation models. Differences in model size, pretraining data, initialization, and computational requirements are reported separately in the experimental section.

3.4 Value architectures

The value model estimates the quality of the current state relative to the task objective. It does not generate low-level actions and does not replace the policy controller. Its outputs are used for state ranking, early failure analysis, trajectory filtering, and ACP-label construction.

We evaluate two value-model variants.

3.4.1 ValueLite

ValueLite is a lightweight multimodal value model designed for rapid training and inference. It combines visual features from the global and wrist cameras with proprioceptive features and task information. Let $f_R(\cdot)$ denote the visual encoder, $f_s(\cdot)$ denote the proprioceptive projection network, and $e(c)$ denote the task embedding. The fused representation is

$$h_t = \text{LN} \left(\left[f_R(I_t^g), f_R(I_t^w), f_s(q_t, g_t), e(c) \right] \right), \quad (2)$$

where LN denotes LayerNorm and $[\cdot]$ denotes feature concatenation.

In the revised implementation, the exact visual backbone, frozen or trainable status, feature dimension, and optimization configuration are fixed across all runs and reported in the reproducibility appendix. These details must not be described ambiguously as “ResNet18/34/50.”

3.4.2 ValueVision

ValueVision uses two visual encoders to examine the effect of geometric and semantic visual representations on state quality estimation. Let $f_D(\cdot)$ denote the DINOv2 encoder and $f_s(\cdot)$ denote the SigLIP encoder. Their features are projected into a common latent space:

$$v_t = W_p \left[f_D(I_t^g, I_t^w), f_s(I_t^g, I_t^w) \right], \quad (3)$$

where W_p is a learnable projection matrix. The projected visual representation is then combined with proprioceptive and task features:

$$h_t = \text{LN} \left(\left[v_t, f_s(q_t, g_t), e(c) \right] \right). \quad (4)$$

DINOv2 is used to provide features related to object geometry and spatial structure, whereas SigLIP provides language-aligned visual features. The exact encoder versions, parameter counts, frozen layers, and feature dimensions are reported in the experimental configuration. Both value architectures use the same target construction, output interface, training split, and evaluation metrics. Their primary difference is the visual representation and associated computational cost.

The two value models have substantially different parameter scales. ValueLite contains 11.462M parameters, of which 0.276M are trainable under the frozen-ResNet18 configuration. ValueVision contains 1185.766M parameters, including the frozen DINOv2-large and SigLIP-SO400M encoders, while 0.683M parameters are trainable

Table 1 – Comparison of the value architectures used in this study. Parameter counts are measured from the checkpoints used for the reported experiments.

Model	Visual Backbone	Input Features	Value Head and Output	Total Params.	Trainable Params.	Role in This Study
ValueLite	Frozen ResNet18	Two RGB views, robot joint positions, gripper state, and task embedding	MLP fusion head with a 201-bin distributional value output	11.462M	0.276M	Low-cost state quality estimation and ACP-label generation
ValueVision	Frozen DINOv2-large and SigLIP-SO400M encoders	Geometric visual features, language-aligned visual features, robot state, and task embedding	Feature projection and multimodal fusion followed by a 201-bin distributional value output	1185.766M	0.683M	Fine-grained visual state evaluation and early failure analysis

in the reported configuration. Therefore, the comparison is interpreted as a comparison between two value-model configurations with different representation capacities and computational costs, rather than as a parameter-matched comparison.

3.5 Value target construction

The proposed target is a task-progress and failure-risk surrogate rather than a universally valid Markovian return. It is designed to transform episode-level success or failure information into a frame-level supervision signal.

Let T_e denote the length of episode e , t denote the current time step, and $T_{\max,c}$ denote the maximum episode length for task c . Let $m_e \in \{0, 1\}$ indicate whether episode e is unsuccessful. We first define an unnormalized episode-dependent target:

$$G_t^{(e)} = -(T_e - t - 1) - m_e c_{\text{fail}} T_{\max,c}, \quad (5)$$

where c_{fail} is the failure-penalty coefficient.

The first term provides a coarse measure of temporal progress: states closer to the end of an episode receive larger values. The second term penalizes states belonging to unsuccessful episodes. This target does not by itself guarantee that a state is recoverable or that it satisfies the Bellman optimality equation. Its interpretation depends on consistent task termination, episode segmentation, and task-specific normalization.

For each task c , the target is normalized to the interval $[-1, 0]$:

$$y_t^{(e)} = \text{clip}(\mathcal{N}_c(G_t^{(e)}), -1, 0), \quad (6)$$

where $\mathcal{N}_c(\cdot)$ denotes the task-specific normalization function. The exact normalization range, c_{fail} , and $T_{\max,c}$ are reported in the experimental configuration. A value closer to 0 indicates greater estimated task progress under this surrogate target, whereas a value closer to -1 indicates lower progress or higher estimated failure risk. This target has two important limitations. First, episode length is only a proxy for task progress and may not be

monotonic in all manipulation scenarios. Second, failed episodes may contain useful recovery states even though they receive a failure-level penalty. For this reason, intervention and recovery frames are treated separately during ACP construction, and the value model is evaluated using both prediction metrics and downstream policy metrics.

3.6 Distributional value learning

Rather than directly regressing a single scalar, the value model predicts a discrete distribution over B value bins. In the current configuration, $B = 201$. Let y_k denote the center of the k -th bin. The value head produces logits and probabilities:

$$z_t = W_v h_t + b_v, \quad p_t = \text{softmax}(z_t), \quad (7)$$

where $p_{t,k}$ denotes the probability assigned to the k -th bin.

The scalar value used for ranking is the expectation of the predicted distribution:

$$V_\phi(o_t, c) = \sum_{k=1}^B p_{t,k} y_k. \quad (8)$$

For a continuous target y_t , the target is projected onto the two adjacent bin centers. If $y_j \leq y_t \leq y_{j+1}$, the soft target distribution is defined as

$$\tilde{y}_{t,j} = \frac{y_{j+1} - y_t}{y_{j+1} - y_j}, \quad \tilde{y}_{t,j+1} = \frac{y_t - y_j}{y_{j+1} - y_j}, \quad (9)$$

with all other entries equal to zero. The value model is optimized using distributed cross-entropy:

$$\mathcal{L}_{\text{value}} = - \sum_{k=1}^B \tilde{y}_{t,k} \log p_{t,k}. \quad (10)$$

Distributional prediction allows the model to represent uncertainty in the predicted target under visual occlusion, contact variation, and noisy supervision. However, the dispersion of the predicted distribution is treated only as

an auxiliary diagnostic unless it is separately calibrated. It is not assumed that the variance of the output distribution is automatically equivalent to execution risk.

The value model is evaluated using value MAE, success-failure separation, early failure AUC, lead time, ACP precision, calibration measures where applicable, and computational cost. These metrics are reported jointly because a lower scalar prediction error alone does not establish that a value model produces useful policy-improvement signals.

3.7 Advantage estimation

The value estimates are used to calculate a local advantage score for each training frame. For a state o_t , we use an n -step estimate:

$$A_t = \sum_{i=0}^{n-1} \gamma^i r_{t+i} + \gamma^n V_\phi(o_{t+n}, c) - V_\phi(o_t, c), \quad (11)$$

where r_t is the task-progress reward, γ is the discount factor, and n is the temporal estimation horizon. The current implementation uses $n = 50$. The corresponding physical time horizon is $n\Delta t$, where Δt is the control interval and is reported in the experimental configuration. The reward definition and discount factor are fixed before evaluation and are not tuned on the evaluation set. If an undiscounted estimate is used, γ is set to 1 and this setting is reported explicitly.

Advantage scores are computed only on training data. For each task, positive ACP samples are selected using a predefined upper-quantile threshold. The current threshold is the top 30% of advantage scores, while threshold sensitivity is evaluated using additional settings in the ablation study.

To avoid incorrectly discarding useful recovery behavior, frames inside manually annotated intervention segments may be assigned a positive recovery label when the corresponding annotation flag is set:

$$\ell_t^{\text{ACP}} = \begin{cases} 1, & \text{if } A_t \text{ exceeds the task-specific threshold,} \\ 1, & \text{if } t \text{ belongs to an annotated recovery segment,} \\ 0, & \text{otherwise.} \end{cases} \quad (12)$$

The intervention annotation is used only for training-label construction. It is not available as ground-truth information during autonomous evaluation. The revised experiments also include an ablation without forced positive intervention labels to measure their independent effect.

3.8 Advantage-conditioned policy

The ACP label is injected into the language condition of SmolVLA. For a task description c , the two training conditions are

$$c_t^+ = c \parallel \text{Advantage: positive}, \quad (13)$$

$$c_t^- = c \parallel \text{Advantage: negative}, \quad (14)$$

where $\parallel$ denotes text concatenation.

The ACP label is used as an auxiliary condition for behavior cloning or flow-matching training. The policy is not optimized using a direct high-variance reinforcement-learning gradient. During training, ACP-label dropout is applied with a predefined probability so that the policy does not depend exclusively on the explicit label.

The dropout probability is fixed before evaluation and is reported in the hyperparameter table. Its effect is evaluated through a dedicated ablation that compares the default dropout setting with no dropout and alternative probabilities.

At deployment time, the evaluation protocol specifies whether the ACP condition is generated by the value model, replaced by a neutral condition, or omitted. Human intervention annotations are never supplied to the policy during autonomous evaluation. This distinction is necessary to prevent intervention information from leaking into the test-time policy input.

3.9 Data-controlled policy retraining

The training data are divided into four episode-level subsets:

- $\mathcal{D}_{\text{demo}}$: successful human teleoperation demonstrations;
- $\mathcal{D}_{\text{rollout}}$: closed-loop robot executions containing distribution-shift and failure states;
- $\mathcal{D}_{\text{recovery}}$: human intervention and recovery trajectories collected after execution drift or failure;
- $\mathcal{D}_{\text{eval}}$: independent real-robot evaluation episodes.

The split is performed at the episode level. Frames from the same episode are not distributed across training and evaluation subsets. The evaluation set is not used to train the policy, value model, ACP threshold, or normalization parameters.

To separate data expansion from value guidance, the retraining variants are defined as follows:

1. **SmolVLA-BC**: trained on $\mathcal{D}_{\text{demo}}$ without ACP;
2. **SmolVLA-Recovery**: trained on $\mathcal{D}_{\text{demo}} \cup \mathcal{D}_{\text{recovery}}$ without ACP;
3. **Data-Expanded BC**: trained on $\mathcal{D}_{\text{demo}} \cup \mathcal{D}_{\text{rollout}} \cup \mathcal{D}_{\text{recovery}}$ without ACP;
4. **SmolVLA+ValueLite**: trained on the same data pool as Data-Expanded BC with ValueLite-generated ACP labels;
5. **SmolVLA+ValueVision**: trained on the same data pool as Data-Expanded BC with ValueVision-generated ACP labels.

The Data-Expanded BC group is necessary because comparing a recovery-only model directly with a value-guided model would confound the value signal with

the additional Rollout-D data. The comparison between SmolVLA-Recovery and Data-Expanded BC measures the effect of adding rollout data. The comparison between Data-Expanded BC and the value-guided variants measures the incremental effect of ACP and value-based selection under a shared data pool.

Additional ablations use random ACP labels, Oracle ACP labels, filtering-only, prompt-only, alternative ACP thresholds, and alternative dropout probabilities. The positive ratio of random labels is matched to the learned ACP labels so that the comparison is not determined by label frequency alone.

All variants use the same SmolVLA backbone, action representation, observation interface, task descriptions, optimizer family, evaluation initialization distribution, and episode-level split. Model-specific parameter counts, trainable parameters, and computational costs are reported separately.

3.10 Training and inference protocol

The policy, value-model, and evaluation protocols are fixed before the final evaluation. The revised manuscript reports the following implementation details:

- policy architecture and parameter count;
- number of trainable and frozen parameters;
- action representation and normalization;
- action horizon and control frequency;
- optimizer, learning rate, scheduler, batch size, and number of epochs;
- image preprocessing and camera synchronization;
- value-backbone versions and trainability;
- value-bin range and target normalization;
- failure-penalty coefficient and reward definition;
- n -step horizon and discount factor;
- ACP threshold and label-dropout probability;
- inference latency, throughput, and peak memory usage.

The evaluation protocol distinguishes autonomous execution from human-assisted execution. An episode is counted as a policy-only success only when the task is completed without human takeover. An assisted success is counted when completion occurs after an intervention. An intervention rate is computed as the fraction of episodes containing at least one human takeover. Recovery success is computed over intervention episodes and measures whether the policy can return to a successful task trajectory after correction.

4. EXPERIMENTS

4.1 Experimental setup and scope

All physical experiments use a LeRobot SO-101 follower arm controlled through the Evo-RL software stack. A second SO-101 arm is used as the leader device for teleoperation. The action vector contains five joint commands and one gripper command. A front RGB camera observes the global workspace, whereas a wrist RGB camera ob-

serves the gripper-object contact region. Both streams are recorded at 640×480 pixels and 30 Hz. Each policy receives the same two camera streams, six-dimensional robot state, language instruction, and six-dimensional action interface. Model-specific image resizing and normalization follow the corresponding training configuration. The quantitative evaluation is restricted to block manipulation with the SO-101 embodiment. The reported tasks assess target-block grasping and placement under fixed workspace geometry. A manipulation episode comprises approach, grasp alignment, grasp closure, transport, and release. Errors in an early phase can therefore alter the state distribution encountered in later phases. The conclusions are limited to this controlled platform and do not imply generalization to other robot embodiments, object categories, or unstructured environments.

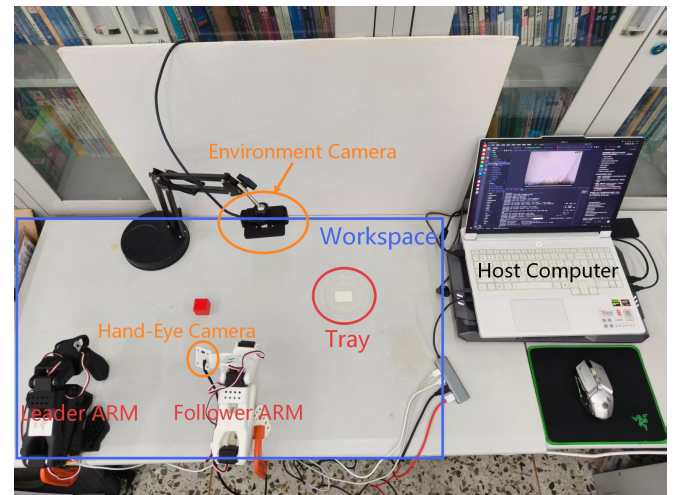


Figure 2 – The real-world robotic evaluation platform and workspace layout. The system relies on a bilateral teleoperation setup consisting of a master leader ARM and a slave follower ARM, overseen by an environment camera, a hand-eye Camera, and a unified host computer processing loop.

4.2 Data composition and controlled comparisons

We distinguish four episode-level data roles. Demo-D contains successful human demonstrations, Rollout-D contains autonomous closed-loop executions, Recovery-D contains human corrective actions after policy drift, and Eval-D contains held-out physical evaluations. Frames from one episode must remain in one partition, and Eval-D must not be used to train policies or value models, select ACP thresholds, or compute normalization statistics. The checkpoint archive used by the reported local training runs references, which contains 101 human-collected successful episodes and 58 971 frames. The archived metadata does not identify separate Rollout-D and Recovery-D episode manifests. Consequently, the additional-data effect cannot be isolated from the current checkpoints alone. We therefore define Data-Expanded BC as a required control rather than assigning it an inferred result. For the controlled experiment, all SmolVLA

Table 2 – Method configurations, training data pools, and experimental roles in the evaluation pipeline.

Method	Policy	Value	Data Pool	ACP	Role
<i>Policy architecture baselines</i>					
ACT-BC	ACT	None	Demo-D	None	Policy architecture baseline
Diffusion-BC	Diffusion Policy	None	Demo-D	None	Policy architecture baseline
SmolVLA-BC	SmolVLA	None	Demo-D	None	VLA policy baseline
<i>Recovery and data-expansion controls</i>					
SmolVLA-Recovery	SmolVLA	None	Demo-D + Recovery-D	None	Recovery-data control baseline
SmolVLA-Expanded BC	SmolVLA	None	Demo-D + Rollout-D + Disabled Recovery-D	Disabled	Data expansion without value guidance
<i>Value-guided policy variants</i>					
SmolVLA+ValueLite	SmolVLA	ValueLite	Demo-D + Rollout-D + Learned Recovery-D	Learned	Value-guided policy variant
SmolVLA+ValueVision	SmolVLA	ValueVision	Demo-D + Rollout-D + Learned Recovery-D	Learned	Value-guided policy variant

variants must start from the same initialization and use the same optimizer, number of updates, batch size, augmentation, and supervised frame set. Data-Expanded BC uses

$$\mathcal{D}_{DE} = \mathcal{D}_{demo} \cup \mathcal{D}_{rollout} \cup \mathcal{D}_{recovery} \quad (15)$$

without value prediction, value-based filtering, advantage labels, or ACP conditioning. Every action-bearing frame is retained, including autonomous rollout actions, so that comparisons with filtering-only and ACP variants can separate sample selection from conditioning. The provenance of each episode and whether its action is autonomous or human-provided must be stored in the released manifest. The compared methods and their data-control conditions are summarized in Table 2. Under this design, SmolVLA-Recovery minus SmolVLA-BC measures the effect of adding recovery supervision; Data-Expanded BC minus SmolVLA-Recovery measures the additional effect of rollout data; and a value-guided variant minus Data-Expanded BC measures the incremental contribution of the value interface under a shared data pool. The last attribution is not made until the Data-Expanded BC row has been evaluated.

4.3 Outcome definitions

Success is determined from the final physical task state rather than from a value-model prediction. Let $y_i = 1$ when episode i completes the specified grasp or placement criterion within the fixed horizon, and let $y_i = 0$ otherwise. The binary outcome variable y_i is determined by the physical task-completion criterion. In the implementation, this annotation is stored in the dataset field; this implementation field is not provided as an input to the deployed policy or the learned value model. The final version must report the physical criterion, the annotation procedure, and the common horizon in steps and seconds.

Let N be the number of evaluation episodes, N_{auto} the number completed without takeover, N_{assist} the number completed after at least one takeover, and N_{int} the number containing at least one takeover. These outcome sets are mutually exclusive, and $N_{succ} = N_{auto} + N_{assist}$. We report

$$\text{SuccessRate} = \frac{N_{succ}}{N}, \quad \text{PolicyOnly} = \frac{N_{auto}}{N}, \quad (16)$$

$$\text{AssistedSuccess} = \frac{N_{assist}}{N}, \quad \text{InterventionRate} = \frac{N_{int}}{N}. \quad (17)$$

If $N_{recovered}$ intervention episodes subsequently satisfy the task-success criterion, recovery success is

$$\text{RecoverySuccess} = \frac{N_{recovered}}{N_{int}}. \quad (18)$$

Thus, the intervention rate uses all evaluation episodes as its denominator, whereas the recovery success uses only intervention episodes. If intervention is disabled for a method, intervention-dependent metrics are reported as not applicable, not as zero. The overall success rate from an intervention-enabled protocol is not interpreted as autonomous policy success. Completion time is averaged over successful episodes. Autonomous and assisted completion times should be reported separately when both are available. For a normalized action vector $\tilde{\mathbf{a}}_t \in [0, 1]^6$, action delta is the mean absolute step-to-step change,

$$\Delta_a = \frac{1}{6(T-1)} \sum_{t=2}^T \|\tilde{\mathbf{a}}_t - \tilde{\mathbf{a}}_{t-1}\|_1. \quad (19)$$

This metric describes command smoothness and is not a substitute for task success.

4.4 Statistical protocol

Binary outcomes are reported as raw counts and Wilson 95% confidence intervals. The current aggregate results contain 50 physical evaluation episodes per method but one training run per checkpoint. They therefore characterize evaluation variation, not training-seed variation. The aggregate counts permit an unpaired two-sided Fisher exact test. A paired McNemar test should replace it if the same initial configurations were used episode by episode and the paired outcomes are retained. The confidence intervals and exploratory pairwise tests are reported in Table 3.

Table 3 – Uncertainty and exploratory pairwise tests for the reported success counts.

Method	Success count	Success rate	Wilson 95% CI
SmolVLA-BC	43/50	86.0%	[73.8%, 93.0%]
SmolVLA-Recovery	45/50	90.0%	[78.6%, 95.7%]
SmolVLA+ValueLite	47/50	94.0%	[83.8%, 97.9%]
SmolVLA+ValueVision	48/50	96.0%	[86.5%, 98.9%]

Pairwise comparison	Absolute difference	Fisher p (two-sided)
BC vs. Recovery	+4 pp	0.760
BC vs. ValueLite	+8 pp	0.318
BC vs. ValueVision	+10 pp	0.160
Recovery vs. ValueLite	+4 pp	0.715
Recovery vs. ValueVision	+6 pp	0.436
ValueLite vs. ValueVision	+2 pp	1.000

Among the SmolVLA recovery and value-guided variants, none of the reported pairwise success-rate differences reaches $p < 0.05$ under the unpaired analysis. Among the SmolVLA recovery and value-guided variants, none of the reported pairwise success-rate differences reaches $p < 0.05$ under the conservative unpaired analysis. Results are therefore described using raw counts, uncertainty intervals, effect directions, and the corresponding test rather than point estimates alone.

4.5 Policy architecture comparison

ACT-BC, Diffusion-BC, and SmolVLA-BC are compared under the demonstration-only condition. The comparison uses the same observation and action interface but is not parameter matched. Model size and training configuration are therefore reported alongside task outcomes.

The phase-level policy results are shown in Table 4.

SmolVLA-BC obtains the highest phase-level success rates in the reported SO-101 evaluation. Its macro action delta is 0.055, compared with 0.069 for ACT-BC and 0.088 for Diffusion-BC. This result indicates smoother normalized commands in this setting, but it does not establish that SmolVLA is universally preferable across embodiments or task distributions. Because the table reports grasp and place phases, it should not be interpreted as an independent test of cross-task or cross-embodiment generalization.

Table 4 – Phase-level policy comparison on the SO-101 block-manipulation setup.

Phase	Method	Success $\uparrow$	Time (s) $\downarrow$	Action Delta $\downarrow$
Grasp	ACT-BC	76.0%	25.4	0.067
	Diffusion-BC	40.0%	47.3	0.084
	SmolVLA-BC	88.0%	16.8	0.052
Place	ACT-BC	72.0%	4.6	0.071
	Diffusion-BC	24.0%	12.5	0.092
	SmolVLA-BC	84.0%	6.2	0.057
Macro average	ACT-BC	74.0%	15.0	0.069
	Diffusion-BC	32.0%	29.9	0.088
	SmolVLA-BC	86.0%	11.5	0.055

4.6 Value architecture comparison

ValueLite and ValueVision use the same 201-bin target range and ACP positive ratio. Value MAE measures absolute target error. The success–failure gap is the difference between mean predictions on successful and failed states. Early failure AUC measures ranking performance inside the fixed pre-failure window. Lead time measures the interval between the first threshold crossing and the annotated failure or takeover. ACP precision is the fraction of positive ACP labels aligned with independently annotated progress or recovery segments. The pre-failure window, threshold, and annotation protocol must be fixed before evaluation and reported with the released metric code. The value-model results are reported in Table 5.

Table 5 – Value-model prediction, separation, early-warning, and ACP-label metrics.

Model	MAE $\downarrow$	S-F $\uparrow$	AUC $\uparrow$	Lead $\uparrow$	ACP P $\uparrow$	Params
ValueLite	0.138	0.421	0.812	2.1	78.0%	11.462M
ValueVision	0.096	0.537	0.884	3.4	86.0%	1185.766M

ValueVision has lower MAE and higher separation, early-warning, and ACP precision point estimates than ValueLite. The comparison is not parameter-matched: ValueVision contains approximately 1.186B parameters, whereas ValueLite contains 11.462M. The predictive results must therefore be interpreted jointly with trainable parameter count and deployment cost.

4.7 Recovery outcomes and value-guided re-training

The intervention-enabled variants have higher total completion point estimates than SmolVLA-BC, but their autonomous outcomes are lower: SmolVLA-BC completes 43/50 episodes without assistance, compared with 40/50 for ValueLite and 41/50 for ValueVision. The 47/50 and 48/50 totals for the value-guided variants therefore reflect assisted recovery as well as autonomous execution. Relative to SmolVLA-Recovery, ValueLite and ValueVision show lower intervention-rate and higher recovery-success point estimates, but the confidence intervals are wide and the aggregate Fisher tests do not establish statistical significance. The autonomous and intervention-assisted outcomes are reported in Table 6.

Table 6 – Autonomous and intervention-assisted outcomes on the real-robot evaluation set.

Method	N	Success	Policy-only	Assisted	Intervention	Recovery	Time (s)
SmolVLA-BC	50	43/50 (86.0%)	43/50 (86.0%)	N/A [†]	N/A [†]	N/A	11.5
SmolVLA-Recovery	50	45/50 (90.0%)	39/50 (78.0%)	6/50 (12.0%)	11/50 (22.0%)	6/11 (54.5%)	13.8
SmolVLA+ValueLite	50	47/50 (94.0%)	40/50 (80.0%)	7/50 (14.0%)	9/50 (18.0%)	7/9 (77.8%)	13.0
SmolVLA+ValueVision	50	48/50 (96.0%)	41/50 (82.0%)	7/50 (14.0%)	8/50 (16.0%)	7/8 (87.5%)	12.6

The Data-Expanded BC checkpoint and its 50-episode evaluation were not found in the experiment archive. Its result is consequently reported as not evaluated rather than estimated from neighboring methods. Until this control is run, the current results cannot determine whether the observed recovery outcomes arise from value-derived ACP labels, additional data, or their interaction.

4.8 Model scale, training configuration, and deployment cost

Table 8 – Model scale and recorded training configuration.

Model	Total	Trainable	Checkpoint	Batch	Steps
ACT-BC	51.669M	51.669M	197.1 MiB	8	100k
Diffusion-BC	266.623M	266.623M	1017.1 MiB	8	100k
SmolVLA	450.046M	99.882M	1142.2 MiB	16	50k
ValueLite	11.462M	0.276M	43.7 MiB	64	200k
ValueVision	1185.766M	0.683M	≈2.22 GiB	8	20k

Parameter counts are computed from the checkpoints used by the reported runs. The models were trained on different GPUs, so wall-clock time and throughput are not compared. Batch-one inference latency and peak runtime memory were not recorded under a common benchmark and are therefore marked as not measured. Parameter count and checkpoint size are reported without using them as substitutes for latency. The model-scale comparison is presented in Table 8, and the checkpoint-level training hyperparameters are listed in Table 7.

4.9 Relation to recent VLA architectures

The scope of the local comparison relative to recent VLA families is summarized in Table 9.

The π_0 family is included as an architectural reference because it is a recent flow-matching VLA family. It is not inserted into the local numerical ranking because the reported study does not contain a compatible SO-101 checkpoint trained and evaluated under the same camera, action, and episode protocol. This exclusion criterion applies equally to other external foundation policies and prevents cross-embodiment pretraining differences from being attributed to the proposed value interface.

4.10 Failure analysis and empirical limits

Observed failures are grouped into approach misalignment, pushing before grasp closure, object slippage, wrist-camera occlusion, placement outside the target region, action stagnation, and unsuccessful recovery after takeover. Qualitative timelines should align the physical event, predicted value, ACP label, and intervention frame. Such examples illustrate failure precursors but do not replace aggregate evaluation.

The current evidence is limited by a single robot embodiment, a narrow block-manipulation task distribution, one training run per checkpoint, and the absence of a completed Data-Expanded BC control. The reported results support a comparison of policy behavior, value prediction, and assisted recovery in this setting. They do not yet establish an autonomous policy improvement caused specifically by value guidance.

5. CONCLUSION, LIMITATIONS AND FUTURE WORK

5.1 Conclusion

This paper studies the interaction between policy architectures, auxiliary value models, and value-guided policy retraining for language-conditioned robot manipulation on a physical SO-101 platform. The experimental

Table 7 – Checkpoint-level optimization and architecture hyperparameters.

Model	Learning rate	Weight decay	Precision	Main configuration
ACT-BC	1×10^{-5}	1×10^{-4}	FP32	ResNet18; chunk 100; 4 encoder and 1 decoder layers; KL weight 10
Diffusion-BC	1×10^{-4}	1×10^{-6}	FP32	Horizon 16; 8 executed actions; 100 train diffusion steps; 3 evaluation de-noising steps
SmolVLA	1×10^{-4}	1×10^{-10}	FP32	Chunk 50; 10 flow steps; frozen vision encoder; expert-only training; warm-up 1000
ValueLite	1×10^{-4}	1×10^{-4}	FP32	Frozen ResNet18; 224 ² input; hidden 256; 201 bins; dropout 0.1
ValueVision	5×10^{-5}	1×10^{-4}	BF16	Frozen DINOv2-large and SigLIP-SO400M; 384 ² input; hidden 512; 201 bins

Table 9 – Scope of the local comparison relative to recent robot-policy families.

Family	Action generation	Training regime	Role in this study
ACT	Transformer action chunks	Task-specific imitation	Local policy baseline
Diffusion Policy	Conditional diffusion	Task-specific imitation	Local policy baseline
SmolVLA	VLM-conditioned flow matching	Task-specific fine-tuning	Local VLA policy and ACP recipient
OpenVLA	Autoregressive action tokens	Large-scale pretrained VLA	External foundation-policy reference
π_0 family	Flow-matching action expert	Cross-embodiment pretraining	Recent external VLA reference

design separates policy capability, value-based state assessment, and the transfer of value information through Advantage-Conditioned Policy (ACP) signals.

Under the demonstration-only protocol, ACT-BC, Diffusion-BC, and SmolVLA-BC exhibit different closed-loop behaviors. Among the evaluated checkpoints, SmolVLA-BC obtains the highest phase-level success rate and the lowest macro-averaged action delta in the tested grasping and placement scenarios. These results characterize the evaluated architectures under the local SO-101 protocol and should not be interpreted as a universal ranking across robot embodiments or task distributions.

The value-model comparison shows that ValueVision achieves a lower value MAE, larger success–failure gap, higher early failure AUC, longer lead time, and higher ACP precision than ValueLite in the reported evaluation. This result indicates that the dual visual representation used by ValueVision provides more discriminative state assessment for the tested block-manipulation scenarios. However, the comparison is not parameter matched. ValueVision contains approximately 1.186 billion parameters, whereas ValueLite contains approximately 11.462 million parameters. The predictive results should therefore be considered together with model scale, trainable parameters, and computational cost.

The intervention-enabled experiments show higher total completion rates for the value-guided variants than for SmolVLA-Recovery: 47/50 episodes are completed with ValueLite and 48/50 with ValueVision, compared with 45/50 for SmolVLA-Recovery. Their intervention-rate point estimates are 18.0%, 16.0%, and 22.0%, respectively, while the corresponding recovery-success rates are 77.8%, 87.5%, and 54.5%. These results suggest an association between value-derived ACP signals and improved assisted recovery in the tested setting.

The autonomous results require a separate interpretation. SmolVLA-BC completes 43/50 episodes without intervention, whereas ValueLite and ValueVision complete 40/50 and 41/50 episodes, respectively. Thus, the higher overall success rates of the value-guided variants include human-assisted episodes and do not demonstrate an improvement in policy-only success. Moreover, the confidence intervals are wide, and the reported unpaired Fisher tests do not establish statistically significant differences among the SmolVLA recovery and value-guided variants. The results should therefore be understood as evidence of value-associated recovery behavior rather than as conclusive evidence of autonomous policy improvement.

A complete causal attribution also requires the Data-Expanded BC control, which must use the same Demo-D, Rollout-D, and Recovery-D pool while removing value prediction, value-based filtering, advantage labels, and ACP conditioning. Since this control was not available in the archived experiments, the present results cannot fully separate the effect of value-derived signals from the effect of additional rollout and recovery data. The main outcome of this study is consequently a clearly defined policy–value evaluation protocol and an initial real-robot analysis of policy behavior, value prediction, and intervention-assisted recovery.

5.2 Limitations

The conclusions are subject to several limitations.

First, the physical evaluation is restricted to SO-101 block manipulation. The reported policy results focus on grasping and placement phases and do not provide broad evidence across diverse objects, workspace layouts, robot embodiments, or semantically complex instructions. The results should therefore be viewed as a controlled real-robot case study.

Second, the archived experiment data do not contain a separately versioned Data-Expanded BC checkpoint or its corresponding 50-episode physical evaluation. The effect of adding Rollout-D therefore cannot currently be isolated from the effect of value-derived ACP supervision. The existing comparison with SmolVLA-BC is insufficient to attribute the observed difference exclusively to value guidance.

Third, the reported ACP experiments do not yet isolate all individual mechanisms. Random-ACP, filtering-only, prompt-only, positive-ratio sensitivity, and indicator-dropout sensitivity require dedicated experiments under the same data pool, initialization, optimization budget, and evaluation episodes. Without these controls, the separate effects of sample filtering, prompt conditioning, and label quality remain unresolved.

Fourth, each reported method is represented by one training checkpoint. The confidence intervals quantify variation over physical evaluation episodes but not variation over training seeds. Multiple training seeds, repeated physical trials, and matched initial configurations are required for a stronger estimate of training and execution variability. Paired statistical tests additionally require episode-level outcomes from matched initial states.

Fifth, intervention conditions are not identical across all baselines. Human intervention is disabled for SmolVLA-BC but enabled for SmolVLA-Recovery and the value-guided variants. Accordingly, the intervention rate is defined over all evaluation episodes, whereas recovery success is defined only over intervention episodes. Total success rates under these protocols must not be interpreted as directly comparable autonomous success rates. Finally, ValueLite and ValueVision operate at substantially different model scales and computational costs. A common batch-one benchmark for preprocessing time, forward-pass latency, peak memory, and power consumption remains necessary. Recent VLA families, including π_0 , are discussed as external architectural references but are not included in the local numerical ranking because compatible checkpoints were not evaluated under the same SO-101 protocol.

5.3 Future work

Future work will first complete the data-controlled comparison by training and evaluating Data-Expanded BC on the full Demo-D + Rollout-D + Recovery-D pool without value-derived signals. The released episode manifest will record data provenance, action source, intervention status, and task identity. This experiment will provide the baseline required to estimate the incremental contribution of ACP.

We will then complete the ACP mechanism ablations, including Random-ACP, filtering-only, prompt-only, and the combined value-guided method. The ACP positive ratio, intervention-frame treatment, and indicator-dropout probability will be varied while keeping the data pool, initialization, training budget, and evaluation episodes fixed.

Additional training seeds and repeated physical evaluations will be used together with confidence intervals, appropriate paired or unpaired tests, and effect-size reporting. The task suite will also be expanded to include different object shapes and colors, target locations, occlusion conditions, tray layouts, and multi-stage sorting instructions.

Finally, deployment efficiency will be improved through value-model distillation, feature caching, quantization of frozen visual encoders, and asynchronous value inference. We will also investigate label-free online monitoring, in which value estimates trigger early warnings, replanning, or human assistance without exposing ACP indicators during execution. These experiments will clarify when value models provide information beyond additional data and whether the policy-value interface transfers across tasks, seeds, and hardware configurations.

ACKNOWLEDGEMENT

The authors would like to express their sincere gratitude to the researchers and engineers of the Evo-RL project for providing the foundational software infrastructure and continuous hardware teleoperation support. We also extend our thanks to the members of our laboratory for their insightful discussions and invaluable technical assistance during the physical robotic deployment and multi-camera calibration phases. This work was supported in part by the National Natural Science Foundation or relevant academic research grants.

REFERENCES

- [1] Dean A. Pomerleau. "Alvin: An Autonomous Land Vehicle in a Neural Network". In: *Advances in Neural Information Processing Systems*. Vol. 1. 1989.
- [2] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. "Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware". In: *Robotics: Science and Systems*. 2023. doi: [10.15607/RSS.2023.XIX.025](https://doi.org/10.15607/RSS.2023.XIX.025).
- [3] Cheng Chi, Zhenjie Feng, Siyuan Du, Zhen Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. "Diffusion Policy: Visuomotor Policy Learning via Action Diffusion". In: *The International Journal of Robotics Research* 43.10–11 (2024), pp. 1566–1587. doi: [10.1177/02783649241273668](https://doi.org/10.1177/02783649241273668).
- [4] Hugging Face Robotics Team. "SmolVLA: A Vision-Language-Action Model for Affordable and Efficient Robotics". In: *arXiv preprint arXiv:2506.01844* (2025).
- [5] Moo Jin Kim, Chelsea Sharma, Benjamin Son, et al. "OpenVLA: An Open-Source Vision-Language-Action Model". In: *Conference on Robot Learning*. 2024.
- [6] Octo Model Team. "Octo: An Open-Source Generalist Robot Policy". In: *Robotics: Science and Systems*. 2024.
- [7] NVIDIA Research. "GR00T N1: An Open Foundation Model for Generalist Humanoid Robots". In: *arXiv preprint arXiv:2503.14734* (2025).
- [8] Kevin Black, Noah Brown, Danny Driess, et al. "pi0: A Vision-Language-Action Flow Model for General Robot Control". In: *arXiv preprint arXiv:2410.24164* (2024).
- [9] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. "Advantage-Weighted Regression: Simple and Scalable Off-Policy Reinforcement Learning". In: *Advances in Neural Information Processing Systems*. Vol. 32. 2019.
- [10] Ashvin Nair, Murtaza Dalal, Abhishek Gupta, and Sergey Levine. "Accelerating Online Reinforcement Learning with Offline Datasets". In: *arXiv preprint arXiv:2006.09359* (2020).
- [11] Ashvin Nair, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. "Overcoming Exploration in Reinforcement Learning with Demonstrations". In: *IEEE International Conference on Robotics and Automation*. 2018, pp. 6292–6299. doi: [10.1109/ICRA.2018.8463162](https://doi.org/10.1109/ICRA.2018.8463162).
- [12] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. "Offline Reinforcement Learning with Implicit Q-Learning". In: *International Conference on Learning Representations*. 2022.
- [13] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. "Decision Transformer: Reinforcement Learning via Sequence Modeling". In: *Advances in Neural Information Processing Systems*. Vol. 34. 2021, pp. 15084–15097.

AUTHORS

HONGLIANG WANG (main author) received his B.S. degree in computer science and technology from Harbin University of Science and Technology, and is currently pursuing a Master's degree at the Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences (UCAS). His primary research interests lie at the strategic intersection of embodied AI, world model, and vision-language-action (VLA) models for physical robotic manipulation platforms.

JIN LIU is a researcher with the Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences (HIAS, UCAS).

XIAOLIN ZHU is a faculty member with the College of Electrical and Information Engineering at the Hunan Institute of Engineering, China. His research interests focus on electrical engineering, intelligent control systems, and information processing. He is dedicated to advancing engineering applications and has contributed to multiple academic and industrial research projects in his field.

QIANG FU is a researcher with the Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences (HIAS, UCAS).

YUHUA ZHENG is with the Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences (HIAS, UCAS). His research interests lie in the fields of intelligent systems, computer vision, and machine learning, with a strong focus on advanced robotic applications and technological innovation. He has been actively involved in several state-of-the-art research initiatives and academic collaborations.

WANGMENG ZUO is a full professor with the School of Computer Science and Technology at the Harbin Institute of Technology (HIT), China. He received his Ph.D. degree in computer application technology from HIT in 2007. His research interests include image modeling and inpainting, deep learning, and computer vision, with a particular focus on image restoration, enhancement, and generative AI. He has published over 100 papers in top-tier journals and conferences such as IEEE TPAMI, CVPR, ICCV, ECCV, and NeurIPS. Prof. Zuo regularly serves as an Area Chair for major computer vision and AI conferences (including CVPR, ICCV, and ECCV) and is an Associate Editor for several international journals.

CHANGQING ZOU is an associate professor at the Hangzhou Institute for Advanced Study, University of Chinese Academy of Sciences (HIAS, UCAS). His research interests lie at the intersection of computer graphics, 3D computer vision, and Embodied AI, specifically focusing on 3D reconstruction, sketch-based modeling, and robotic perception. He has published numerous papers in top-tier conferences and journals, including CVPR, ICCV, SIGGRAPH, and TOG.

EVGENY BURNAEV graduated from the Moscow Institute of Physics and Technology (MIPT) in 2006. He holds a Ph.D. in theoretical foundations of informatics from the Institute for Information Transmission Problems (2008) and a Habilitation (Doctor of Science) in mathematical modeling from MIPT (2021). Prof. Burnaev currently serves as a full professor and director of the Skoltech Applied AI Center. He specializes in machine learning (generative modeling and manifold learning), industrial predictive analytics, and 3D computer vision, with various applications spanning engineering optimization, 3D reconstruction, and medical imaging.