ACTION–EFFECT CONTRACTS: BRIDGING TASK INTENT AND PHYSICAL EFFECTS IN AGENTIC ROBOTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action (VLA) models map language instructions and visual observations directly to continuous robot actions. However, Real-world robot tasks are often long-horizon and compositional, requiring agentic robot systems to plan and execute sequences of interdependent subtasks. In mainstream designs, a high-level agent intervenes through task-flow decisions such as subtask invocation, stopping, replanning, and human assistance, while the low-level VLA determines the specific effects of task execution. This architecture leaves the agent without direct involvement in action execution. To increase the agent's participation during execution, we introduce *Action–Effect Contracts*, a semantic Agent–VLA interface that enables the high-level agent specify the physical effect required at the current task stage. At each decision point, a frozen VLA proposes multiple short action candidates, and an effect ranker selects the candidate most likely to realize the requested effect. The ranker jointly conditions on the current image, robot state, effect request, and candidate action through candidate-dependent spatial attention. Same-state counterfactual rollouts provide candidate-level matching labels and within-group ranking supervision. We instantiate the high-level agent as a fine-tuned Qwen3-VL-8B-Instruct model that generates online effect requests, and use a frozen $\pi_{0.5}$ action policy for short-horizon manipulation in LIBERO. Across nine LIBERO-90 tasks, our method improves requested-effect matching in all four stages and raises task success from 42.06% to 51.27%. In four real-world manipulation tasks performed by a robot arm, success increases from 48.33% to 58.33%. These results show that high-level effect requests can effectively guide VLA action execution and improve task completion.

1 INTRODUCTION

Recent vision-language-action (VLA) models have demonstrated strong robot manipulation capabilities by mapping language instructions and visual observations directly to continuous robot actions (Physical Intelligence et al., 2025). However, many real-world goals are long-horizon and compositional: they require a sequence of interdependent subtasks whose execution depends on prior outcomes. Early systems use language models to plan and update subtask sequences executed by low-level skills (Ahn et al., 2022; Huang et al., 2022). Recent agentic systems couple high-level intent interpretation and planning with low-level VLA execution (Shi et al., 2025; Li et al., 2026).

This division of labor makes the boundary between reasoning and execution central (Hu et al., 2026). In common skill-selection and hierarchical-policy designs, the high-level agent selects and invokes a subtask, and the lower-level policy generates the controls needed to execute it (Ahn et al., 2022; Shi et al., 2025). The agent can intervene at coarse task-flow decision points through stopping execution, replanning, or requesting human assistance (Huang et al., 2022; Li et al., 2026). These mechanisms reshape task flow while offering limited control over the specific effects produced during task execution. These designs provide limited explicit support for effect-conditioned selection among low-level action candidates, motivating an interface that increases the agent's participation during VLA execution.

In our implementation, the high-level agent is a fine-tuned Qwen3-VL-8B-Instruct model. To give this agent an effective means to intervene during low-level VLA execution, we propose *Action–Effect*

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

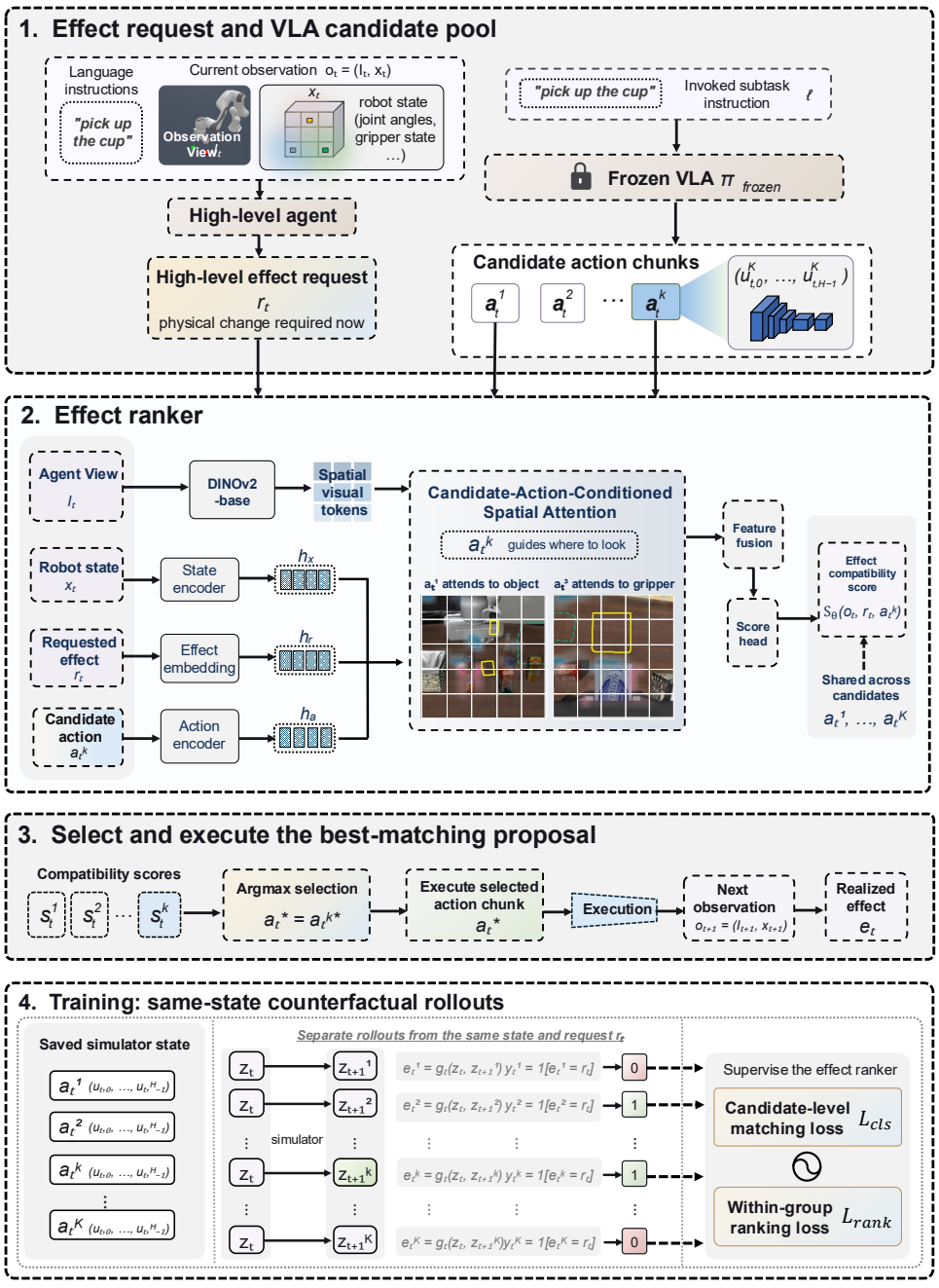


Figure 1: Overview of Action–Effect Contracts. The high-level agent generates an effect request, and the frozen VLA proposes task-conditioned action chunks. The effect ranker combines visual features, robot state, the request, and each candidate through candidate-conditioned spatial attention to score effect compatibility. The highest-scoring chunk is executed. During simulation training, same-state counterfactual rollouts provide effect-matching labels for candidate-level classification and within-group ranking.

Contracts. An Action–Effect Contract provides a semantic interface through which the high-level agent specifies the physical effect required at the current task stage.

We realize this interface through *effect-conditioned action selection*. At each decision point, a frozen VLA proposes multiple short action candidates under the same observation and task instruction. An effect ranker uses the current effect request to select the candidate most likely to realize the requested physical effect. This selection introduces high-level intent into low-level action selection and influences the continuous action sent to the robot. The VLA parameters and action-generation procedure remain unchanged. We evaluate this intervention through agreement between the requested and realized physical effects.

The ranker jointly conditions on the current image, robot state, effect request, and candidate action. Candidate-dependent attention over visual patches allows different proposed motions to use different scene evidence. We train the ranker with counterfactual rollouts of candidate actions from the same saved simulator state, yielding labels that identify whether each candidate realizes the requested effect. The high-level agent generates the online effect requests. We instantiate the action policy with a frozen $\pi_{0.5}$ for short-horizon manipulation in LIBERO (Liu et al., 2023).

We evaluate Action–Effect Contracts on nine LIBERO-90 tasks and four real-world manipulation tasks using an AgileX Piper robot arm. In LIBERO-90, the method improves requested-effect matching across approach, lift, transport, and place, and raises overall task success from 42.06% with direct policy execution to 51.27%. On the AgileX Piper robot arm, success increases from 48.33% to 58.33%. Compared with random selection from the same number of candidates, the gains are 7.62 and 11.67 percentage points in simulation and with the robot arm, respectively. Ablations in LIBERO-90 show the contributions of effect requests, candidate-dependent visual attention, ranking supervision, and robot state; removing the effect request causes the largest success-rate drop of 10.48 percentage points. These findings show that high-level effect requests can effectively guide VLA action execution through candidate selection and improve task completion.

This paper makes three contributions. First, it introduces Action–Effect Contracts as a semantic Agent–VLA interface that addresses limited execution-time coupling by making the physical effect required at each decision point explicit. Second, it develops a candidate-conditioned ranker trained through same-state counterfactual supervision to select actions from a frozen VLA according to these effect requests. Third, it demonstrates improved requested-effect matching in LIBERO-90 and higher task success in both simulation and robot arm manipulation in the real world, with component ablations supporting the design.

2 Related Work

2.1 Vision-Language-Action Models for Robot Manipulation

VLA models map vision and language to robot actions. RT-1 scales language-conditioned control (Brohan et al., 2023b); RT-2 tokenizes actions (Brohan et al., 2023a). Open X-Embodiment contributes cross-robot data and RT-X models (Open X-Embodiment Collaboration et al., 2025); Octo and OpenVLA offer open generalist policies (Octo Model Team et al., 2024; Kim et al., 2024). RDT-1B uses diffusion for bimanual manipulation (Liu et al., 2025), and flow-based π_0 precedes the broader-task $\pi_{0.5}$ (Black et al., 2026; Physical Intelligence et al., 2025). OpenVLA-OFT improves continuous, chunked actions (Kim et al., 2025); GR00T N1 pairs vision-language reasoning with continuous actions (NVIDIA et al., 2025). Embodied Chain-of-Thought adds intermediate reasoning (Zawalski et al., 2025); MolmoAct provides editable spatial trajectories (Lee et al., 2025). We rank frozen $\pi_{0.5}$ proposals using online effect requests.

2.2 Agentic Robot Systems for Long-Horizon Tasks

SayCan selects feasible skills (Ahn et al., 2022); ProgPrompt generates situated plans (Singh et al., 2022); Code as Policies writes reactive programs (Liang et al., 2023). SayPlan replans with 3D scene graphs (Rana et al., 2023); Inner Monologue updates plans from feedback (Huang et al., 2022); OK-Robot combines perception, navigation, and grasping (Liu et al., 2024). Hi Robot connects high-level instructions and feedback to VLA execution (Shi et al., 2025); RoboClaw schedules, monitors, and recovers policies (Li et al., 2026). Planner, policy, and interface choices shape hierarchical VLA performance (Hu et al., 2026).

Within subtasks, VoxPoser plans from 3D value maps (Huang et al., 2023), ReKep optimizes relational keypoint constraints (Huang et al., 2024), and VLMPC selects predicted sequences by goal cost (Zhao et al., 2024). Our interface specifies physical effects online and ranks short frozen-VLA candidates under the same observation.

3 METHODOLOGY

Action–Effect Contracts connect a high-level agent's assessment of the required physical effect to the selection of a short action chunk from a frozen VLA. Our implementation combines a request-generating agent, a candidate-conditioned effect ranker, and supervision from the outcomes of candidate execution. Figure 1 summarizes the execution and training pipeline. We describe the decision variables, request generation, ranker architecture, and training procedure below.

3.1 PROBLEM FORMULATION

Let ℓ denote the language instruction for an active subtask and t a decision round within its execution. The current observation is $o_t = (I_t, x_t)$, where I_t is the scene image and x_t is the robot state. The high-level agent generates a request $r_t \in \mathcal{R}$, with $\mathcal{R} = \{\text{approach}, \text{lift}, \text{transport}, \text{place}\}$. Each request specifies the physical effect required at the current task stage. The agent updates this request after each executed action chunk using the new observation and its recent history.

An action chunk is $a_t = (u_{t,0}, \ldots, u_{t,H-1})$, where H is the number of control steps and $u_{t,h}$ is the robot control at step h. Its execution produces a realized effect $e_t \in \mathcal{E}$, where $\mathcal{E}$ contains the four requested effects and outcomes outside that vocabulary. An Action–Effect Contract pairs the request with the agreement criterion

$$m_t = \mathbf{1}[e_t = r_t], \tag{1}$$

where m_t equals one when the realized label matches the request and zero otherwise. The effect label is assigned from the action-induced physical transition. This agreement criterion supplies the ranker's learning target and the requested-effect Hit metric. Task success measures completion over the full execution sequence.

3.2 EFFECT REQUEST GENERATION

Decision-time inputs and outputs. The high-level agent is Qwen3-VL-8B-Instruct,[1] adapted to generate effect requests through supervised fine-tuning with LoRA. Its input contains the active task instruction, global and wrist-camera RGB images from the preceding and current decision rounds, and the corresponding end-effector poses and gripper states.

The agent emits a short structured response specifying one of the four effects. Approach requests an actionable gripper position relative to the object; lift requests an increase in object height; transport requests movement toward the target; and place requests the target spatial relation. Deterministic decoding supplies the request before candidate selection. After the selected chunk is executed, the new images and robot state update the input for the next request. In simulation, this update occurs after every five control steps. Appendix A gives the output vocabulary and execution procedure.

Request supervision. Agent training pairs each decision-time input with the effect required at that state. Simulation annotations are derived offline from object proximity, object elevation, proximity to the target, and the target spatial relation. Trajectories collected with different action-selection behaviors cover both task progression and states requiring renewed interaction with the object. Supervised fine-tuning applies the language-model loss to the request answer tokens. Training and validation are separated by initial state, and evaluation uses held-out initial states.

For real-world adaptation, human annotators assign the required effect from the task instruction and synchronized visual and robot-state observations. The simulation-adapted agent is further fine-tuned on these request annotations with simulation replay, using scene- or session-level data separation. These labels describe the effect needed before an action is chosen. The ranker's outcome labels, introduced below, describe the physical change produced by executing a candidate.

[1] Base model: `https://huggingface.co/Qwen/Qwen3-VL-8B-Instruct`.

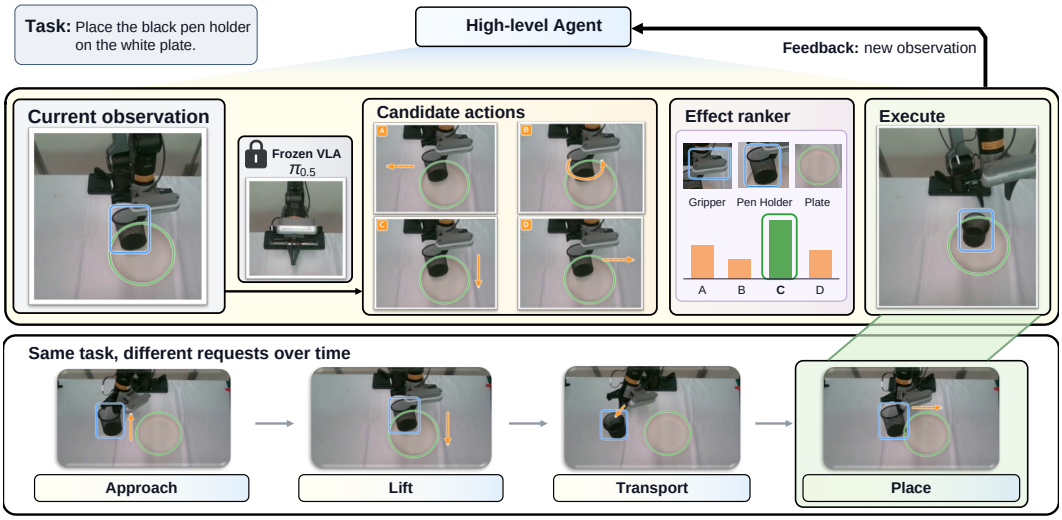


Figure 2: Illustration of request-conditioned candidate selection. The agent specifies the current effect request, the frozen VLA proposes candidate actions, and the effect ranker selects a candidate for execution. Updated observations feed back to the agent as the task progresses. Candidate motions and scores are schematic.

3.3 Effect-Conditioned Action Selection

Candidate generation and encoding. At round t, the frozen VLA policy π_{frozen} samples K candidate chunks under the same observation and instruction:

$$\mathcal{A}_t = (a_t^1, \ldots, a_t^K), \qquad a_t^k \sim \pi_{\text{frozen}}(\cdot \mid o_t, \ell). \tag{2}$$

Here $\mathcal{A}_t$ is the ordered candidate pool and k indexes its candidates. Our simulation implementation uses $\pi_{0.5}$ with $K = 4$ and $H = 5$. The ranker receives the current global image, robot state, effect request, and each candidate's complete control sequence.

The task instruction specifies the manipulated object and destination to $\pi_{0.5}$, which generates task-conditioned action candidates. These candidates, together with the current image, provide the ranker with implicit cues about the object and target. The ranker combines these cues with the current effect request to score the proposed actions.

A frozen DINOv2-base backbone (Oquab et al., 2023) extracts spatial patch features from the image. We retain the patch tokens and project their 768-dimensional features to a shared dimension $d = 256$, giving N patch vectors $v_{t,i} \in \mathbb{R}^d$, where i indexes a patch. Separate state and action encoders each apply a linear layer, GELU, and layer normalization to produce d-dimensional embeddings h_t^x and $h_t^{a,k}$. The action encoder receives the flattened candidate chunk. In simulation, the state has eight dimensions and each chunk contains five seven-dimensional controls. A learned four-entry embedding table maps the request to $h_t^r \in \mathbb{R}^d$. State and action coordinates are standardized using training-set statistics.

Figure 2 illustrates how an effect request guides candidate scoring within the execution loop.

Candidate-dependent visual evidence. The state and request form a shared context $c_t = h_t^x + h_t^r$. Each candidate contributes its own action embedding to the attention query:

$$q_t^k = \mathrm{LN}(h_t^{a,k} + c_t), \qquad \alpha_{t,i}^k = \frac{\exp((q_t^k)^\top v_{t,i}/\sqrt{d})}{\sum_{j=1}^N \exp((q_t^k)^\top v_{t,j}/\sqrt{d})}, \qquad \bar{v}_t^k = \sum_{i=1}^N \alpha_{t,i}^k v_{t,i}. \tag{3}$$

Here LN is layer normalization, q_t^k is the candidate query, $\alpha_{t,i}^k$ is its attention weight for patch i, j runs over image patches, and $\bar{v}_t^k$ is the resulting visual summary. The image features are shared across candidates, while their queries and visual summaries depend on the proposed controls. This

allows the ranker to evaluate each motion using scene evidence selected jointly by that motion, the robot state, and the requested effect.

The scoring head concatenates $\bar{v}_t^k$, $h_t^{a,k}$, and c_t into a 768-dimensional vector, then applies a 256-unit linear layer, GELU, dropout with rate 0.1, and a scalar output layer. Its output $s_t^k = s_\theta(o_t, r_t, a_t^k)$ is a compatibility logit, where θ comprises the trainable projection, embeddings, encoders, query normalization, and scoring head. Directly retaining the action embedding in this head lets the score use the full proposed motion together with its visual summary.

Selection and execution. The robot executes the maximum-score candidate:

$$k_t^* = \arg\max_{k \in \{1,\dots,K\}} s_t^k, \qquad a_t = a_t^* = a_t^{k_t^*}. \tag{4}$$

Here k_t^* is the selected index; the chosen candidate a_t^* becomes the executed chunk a_t. Its observed effect e_t determines m_t through Equation 1. Selection uses the relative scores across the complete pool, including rounds in which all scores are low or several are high; ties follow candidate order. An abstention or unparseable agent response triggers uniform candidate selection for that round. After execution, the agent receives the updated observation and generates the next request. The VLA and visual backbone remain frozen throughout this process.

3.4 Training and Supervision

Same-state candidate outcomes. Training examples form candidate groups indexed by b, each sharing an observation, task instruction, and effect request. In simulation, z_b is the group's saved starting state, restored before each candidate a_b^k. Let $z_b^{k,\mathrm{end}}$ denote the state after all H control steps and τ_b the group's task specification identifying the object and target. The annotation rule g_{τ_b} assigns an effect label and a binary matching target:

$$e_b^k = g_{\tau_b}(z_b, z_b^{k,\mathrm{end}}), \qquad y_b^k = \mathbf{1}[e_b^k = r_b]. \tag{5}$$

Here r_b is group b's request, e_b^k is candidate k's realized effect, and y_b^k indicates their match. Executing candidates from a shared starting state connects differences in their controls to differences in physical outcomes.

Simulation labels use changes across the chunk. Approach requires at least a $1\,\mathrm{cm}$ decrease in end-effector–object distance. Lift requires an object-height increase of at least $1\,\mathrm{cm}$ with endpoint end-effector–object distance at most $8\,\mathrm{cm}$. Transport requires at least a $1\,\mathrm{cm}$ decrease in object–target distance, and place requires the endpoint to satisfy the task's target spatial relation. When several conditions hold, the priority place, transport, lift, approach assigns one label; transitions satisfying none receive an outcome outside $\mathcal{R}$. Appendix B details this annotation scheme.

The real-world ranker is trained on physical execution data with human-annotated effect outcomes. These data include groups of candidate actions executed from the same starting conditions. Each candidate's annotated outcome is compared with the group's effect request to obtain its matching target, retaining the same candidate-level and within-group supervision used in simulation.

Matching and ranking objectives. Let $\mathcal{B}$ index the candidate groups in a training batch and $s_b^k = s_\theta(o_b, r_b, a_b^k)$ denote candidate k's score under group observation o_b and request r_b. The weighted binary classification loss is

$$\mathcal{L}_{\mathrm{cls}} = -\frac{1}{K|\mathcal{B}|} \sum_{b \in \mathcal{B}} \sum_{k=1}^{K} \left[w_+ y_b^k \log \sigma(s_b^k) + (1 - y_b^k) \log\bigl(1 - \sigma(s_b^k)\bigr) \right], \tag{6}$$

where $|\mathcal{B}|$ is the batch's group count, σ is the sigmoid function, and w_+ is the training-set ratio of negative to positive candidate labels. This term supervises every candidate, with positive weighting accounting for the frequency of matching outcomes.

Let $\mathcal{P}_\mathcal{B}$ contain all triples (b, i, j) with $b \in \mathcal{B}$ for which candidate i matches the request and candidate j from the same group does not. The within-group ranking loss and total objective are

$$\mathcal{L}_{\mathrm{rank}} = \frac{1}{|\mathcal{P}_\mathcal{B}|} \sum_{(b,i,j) \in \mathcal{P}_\mathcal{B}} \log\Bigl(1 + \exp(s_b^j - s_b^i)\Bigr), \qquad \mathcal{L} = \mathcal{L}_{\mathrm{cls}} + \mathcal{L}_{\mathrm{rank}}. \tag{7}$$

Here $|\mathcal{P}_{\mathcal{B}}|$ is the number of within-group positive–negative pairs; the ranking term is zero for a batch with no such pairs. Classification teaches whether an individual candidate realizes a request, while ranking explicitly raises matching candidates above nonmatching alternatives available at the same decision point. The two terms have equal weight. We optimize the trainable ranker components with AdamW, keeping candidate groups together in each training batch and freezing the VLA and DINOv2 backbone.

4 EXPERIMENTS

We evaluate task success, requested-effect matching, and ranker components on nine LIBERO-90 and four real-world robot arm manipulation tasks. Appendix C expands the evaluation protocol and ablation definitions, and Appendix D provides supplementary analysis and execution visualization.

4.1 EXPERIMENTAL SETUP

Simulation protocol. Nine LIBERO-90 kitchen and living-room tasks cover grasping, lifting, transporting, and placing (Liu et al., 2023). We select grasping-based tasks whose success rate under the original $\pi_{0.5}$ policy is above 10% and below 90%. Each method runs 70 episodes per task (630 total) with shared initial states, control-step budgets, and a frozen $\pi_{0.5}$ checkpoint. The high-level agent, a fine-tuned Qwen3-VL-8B-Instruct model, generates each effect request; selection uses $K = 4$ candidate chunks of $H = 5$ control steps.

Compared methods. **Default** executes the policy's default chunk; **Random** selects uniformly from four proposals; **Ours** executes the proposal with the highest effect-conditioned score. Random shares Ours' candidate budget, while Default measures direct policy execution.

Metrics. Success rate (SR) is the fraction of episodes reaching the goal within the control-step budget. Hit is the fraction of executed chunks matching the request (Equation 1), pooled by method and request across all nine tasks. Rounds with an abstention or an unparseable request are excluded from the Hit denominator. Simulator transitions assign labels: *approach* reduces end-effector–object distance by $\geq 1\,\mathrm{cm}$; *lift* raises the object by $\geq 1\,\mathrm{cm}$ and ends within $8\,\mathrm{cm}$ of the end effector; *transport* reduces object–target distance by $\geq 1\,\mathrm{cm}$; and *place* satisfies the target spatial relation. Ties resolve in the order place, transport, lift, approach. Rates are percentages; differences are percentage points (pp).

4.2 TASK SUCCESS IN LIBERO-90

Table 1: LIBERO-90 task success (70 episodes per task and method). Best results in bold.

Method	rear black bowl onto plate	middle black bowl onto plate	black bowl into upper drawer	black bowl onto plate	black bowl onto cabinet top	right moka pot onto stove	white bowl onto cabinet top	ketchup bottle into tray	left black bowl into tray	Overall
Default	14.29	77.14	70.00	62.86	24.29	15.71	17.14	44.29	52.86	42.06
Random	15.71	74.29	71.43	65.71	22.86	30.00	17.14	45.71	50.00	43.65
Ours	**21.43**	**84.29**	**85.71**	**67.14**	**31.43**	**34.29**	**20.00**	**54.29**	**62.86**	**51.27**

Ours reaches 51.27% overall SR versus 42.06% for Default and 43.65% for Random (Table 1), gains of 9.21 and 7.62 pp. It leads on all nine tasks. Gains over Random range from 1.43 pp for black-bowl placement onto a plate to 14.29 pp for placement into the upper drawer, extending across different destinations for similar objects.

Random exceeds Default on five tasks, falls below it on three, and ties on one. Ours' consistent gains support effect-conditioned selection (Section 3.3): with the VLA frozen and the same candidate budget as Random, using the agent's request to score and select proposals improves task completion.

4.3 REQUESTED-EFFECT MATCHING

Table 2: Requested-effect Hit across LIBERO-90, pooling action chunks by method and request.

Method	Approach	Lift	Transport	Place
Default	26.02	16.20	68.89	29.51
Random	24.91	15.93	66.40	36.29
Ours	**34.21**	**24.04**	**78.34**	**44.96**

Ours improves Hit over Random by 9.30 pp for approach, 8.11 for lift, 11.94 for transport, and 8.67 for place (Table 2). Transport shows the largest absolute gain, while lift rises from 15.93% to 24.04%, an approximately 51% relative increase. Improvements therefore extend to both a frequently matched effect and one with a lower baseline Hit.

Random improves place Hit over Default but lowers the other three rates. Effect-conditioned selection improves all four, supporting the request–outcome agreement objective in Section 3.1. Hit measures executed transitions, linking candidate scoring to realized physical changes; Table 1 connects this action-level matching objective to task completion.

4.4 ABLATION OF THE EFFECT RANKER

Table 3: Effect-ranker ablations on nine LIBERO-90 tasks. Stage columns report Hit; SR reports task success. All ranker variants share the frozen policy and candidate budget.

Variant	Approach	Lift	Transport	Place	SR
Default	26.02	16.20	68.89	29.51	42.06
Full ranker (Ours)	**34.21**	**24.04**	**78.34**	**44.96**	**51.27**
Without effect request	26.92	16.41	63.10	31.12	40.79
Global visual pooling	27.84	18.66	71.17	31.93	42.54
Without ranking loss	29.82	20.39	72.41	35.59	43.65
Without robot state	29.45	22.58	73.59	37.37	47.46

Each ablation changes one component under the same evaluation protocol (Table 3). Removing the effect request produces the largest SR decrease, 10.48 pp, and reduces transport Hit by 15.24 pp. Section 3.3 incorporates the request into both the attention query and scoring context. These losses support using the agent's current requirement to determine which candidate effects are relevant to execution.

Global visual pooling reduces SR by 8.73 pp and place Hit by 13.03 pp, its largest stage-wise drop. The full model retrieves spatial evidence through action-, state-, and request-dependent queries. This comparison supports candidate-specific visual summaries for assessing whether a motion achieves the required placement relation.

Removing ranking supervision reduces SR by 7.62 pp and lowers all four Hit rates. Candidate-level classification is retained, so the comparison supports the additional within-group objective in Section 3.4: learning to order matching candidates above nonmatching alternatives directly serves the maximum-score execution rule.

Removing robot state yields a smaller SR decrease of 3.81 pp, with its largest Hit drop in place (7.59 pp). State enters both the visual query and scoring context, providing information about the robot's configuration alongside the image and proposed controls. This result supports their joint use when evaluating candidate outcomes.

4.5 REAL-WORLD MANIPULATION

On the AgileX Piper robot arm, each method attempts to place a black pen holder, eggplant, corn cob, and banana into a white plate, with 30 trials per object. Success requires releasing the object and

Table 4: Task success with the AgileX Piper robot arm (30 trials per object and method; 120 per method overall).

Method	Place the black pen holder in the white plate	Place the eggplant in the white plate	Place the corn cob in the white plate	Place the banana in the white plate	Overall
Default	33.33	50.00	53.33	56.67	48.33
Random	36.67	46.67	43.33	60.00	46.67
Ours	**46.67**	**56.67**	**60.00**	**70.00**	**58.33**

leaving it stably in the plate. Following Section 3.4, the ranker uses real-world execution data with human-annotated outcomes, including candidates executed from the same starting conditions.

Ours reaches 58.33% SR, exceeding Default by 10.00 pp and Random by 11.67 pp (Table 4). Relative to Random, corn placement gains 16.67 pp, while each remaining task gains 10.00 pp. Random falls below Default on eggplant and corn, whereas Ours improves both, supporting learned effect-conditioned evaluation of the available proposals. With real-world outcome supervision, the interface also improves physical task completion, extending the simulation evidence for agent participation through candidate selection.

5 Conclusion

We introduced Action–Effect Contracts, a semantic Agent–VLA interface that increases the high-level agent's participation in VLA action execution through physical-effect requests. The high-level agent, a fine-tuned Qwen3-VL-8B-Instruct model, generates the physical-effect request at each decision point, and a learned ranker uses this request to select among short action chunks proposed by a frozen $\pi_{0.5}$ policy. Candidate-dependent spatial attention and same-state counterfactual supervision enable the ranker to associate proposed actions with their physical effects, allowing the agent's intent to guide execution throughout an ongoing subtask.

Across nine LIBERO-90 tasks, the method improves requested-effect matching in all four stages and increases overall task success from 42.06% to 51.27%. In four real-world manipulation tasks performed by a robot arm, success increases from 48.33% to 58.33%. The gains over random candidate selection are 7.62 and 11.67 percentage points in simulation and with the robot arm, respectively. Ablations show the contributions of effect requests, candidate-dependent visual attention, ranking supervision, and robot state to these improvements. These results show that high-level effect requests can effectively influence the actions executed by the VLA through effect-conditioned selection, improving requested-effect matching and task completion.

6 Limitations and Future Work

The current implementation is adapted only to grasping-based manipulation tasks. Future work will adapt the framework to a broader range of manipulation tasks, including pushing, pulling, and pressing. A local effect may also change later subtask feasibility. Future work can connect immediate effects, subtask progress, and terminal goals through temporal contract hierarchies and long-horizon value estimates.

Our method selects actions from candidates generated by a frozen VLA, so its effectiveness depends on the policy's capabilities and the coverage and quality of its training data. When the candidate pool lacks actions capable of producing the requested effect, selection alone has limited scope for improvement. Future work will explore stronger VLA policies and broader manipulation data to improve candidate quality and expand the range of achievable effects. To address requests that exceed the policy's action distribution, predicted candidate effects could assess reachability and guide request revision, candidate generation, or intermediate-effect planning.

AI USE STATEMENT

We used AI tools to assist with polishing the manuscript, including improving sentence clarity, readability, and consistency of expression while preserving the intended meaning. These tools also supported checks for grammatical and spelling errors. In addition, we used AI assistance to refine the manuscript's formatting and layout, including the placement of figures and tables, text wrapping, and page organization, to improve the presentation of the paper.

ETHICS STATEMENT

This work studies agent-guided robotic manipulation through simulation and object-placement tasks on a robot arm. Its intended contribution is to make action execution more responsive to explicitly stated physical-effect requests. Responsible deployment should include task-appropriate safety evaluation, human oversight, and safeguards against unintended contact and object damage. Reuse of pretrained models and datasets should respect their licenses and usage conditions, while collection of real-world observations should protect the privacy of people in the workspace.

REPRODUCIBILITY STATEMENT

Section 3 describes the agent inputs, effect-ranker architecture, candidate selection rule, and training objectives. Appendices A and B specify request generation, execution procedures, effect annotation, and candidate-group supervision. Section 4 and Appendix C report task selection, comparison methods, evaluation counts, success criteria, Hit aggregation, and ablation definitions. Simulation uses a frozen $\pi_{0.5}$ policy with four candidates of five control steps each. Real-world ranker training uses human-annotated execution outcomes, including candidate groups with shared starting conditions. These details support reproducing the learning and evaluation procedures and interpreting the reported comparisons.

REFERENCES

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do as i can, not as i say: Grounding language in robotic affordances, 2022. URL https://arxiv.org/abs/2204.01691.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2026. URL https://arxiv.org/abs/2410.24164.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-language-action models transfer web knowledge to robotic control, 2023a. URL https://arxiv.org/abs/2307.15818.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics transformer for real-world control at scale, 2023b. URL https://arxiv.org/abs/2212.06817.

Jiaheng Hu, Mohit Shridhar, Caden Lu, Dhruv Shah, Hao-Tien Lewis Chiang, Jie Tan, and Annie Xie. What matters in orchestrating robot policies: A systematic study of hierarchical VLA agents, 2026. URL https://arxiv.org/abs/2606.10267.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models, 2022. URL https://arxiv.org/abs/2207.05608.

Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. VoxPoser: Composable 3D value maps for robotic manipulation with language models, 2023. URL https://arxiv.org/abs/2307.05973.

Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. ReKep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation, 2024. URL https://arxiv.org/abs/2409.01652.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model, 2024. URL https://arxiv.org/abs/2406.09246.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success, 2025. URL https://arxiv.org/abs/2502.19645.

Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, Winson Han, Wilbert Pumacay, Angelica Wu, Rose Hendrix, Karen Farley, Eli VanderBilt, Ali Farhadi, Dieter Fox, and Ranjay Krishna. MolmoAct: Action reasoning models that can reason in space, 2025. URL https://arxiv.org/abs/2508.07917.

Ruiying Li, Yunlang Zhou, YuYao Zhu, Kylin Chen, Jingyuan Wang, Sukai Wang, Kongtao Hu, Minhui Yu, Bowen Jiang, Zhan Su, Jiayao Ma, Xin He, Yongjian Shen, Yang Yang, Guanghui Ren, Maoqing Yao, Wenhao Wang, and Yao Mu. RoboClaw: An agentic framework for scalable long-horizon robotic tasks, 2026. URL https://arxiv.org/abs/2603.11558.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control, 2023. URL https://arxiv.org/abs/2209.07753.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 2023. URL https://arxiv.org/abs/2306.03310.

Peiqi Liu, Yaswanth Orru, Jay Vakil, Chris Paxton, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Demonstrating OK-Robot: What really matters in integrating open-knowledge models for robotics. In *Robotics: Science and Systems XX*, RSS2024. Robotics: Science and Systems Foundation, July 2024. doi: 10.15607/rss.2024.xx.091. URL http://dx.doi.org/10.15607/RSS.2024.XX.091.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1B: a diffusion foundation model for bimanual manipulation, 2025. URL https://arxiv.org/abs/2410.07864.

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy, 2024. URL https://arxiv.org/abs/2405.12213.

Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anchit Gupta, Andrew Wang, Andrey Kolobov, Anikait Singh, Animesh Garg, Aniruddha Kembhavi, Annie Xie, Anthony Brohan, Antonin Raffin, Archit Sharma, Arefeh Yavary, Arhan Jain, Ashwin Balakrishna, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Blake Wulfe, Brian Ichter, Cewu Lu, Charles Xu, Charlotte Le, Chelsea Finn, Chen Wang, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Christopher Agia, Chuer Pan, Chuyuan Fu, Coline Devin, Danfei Xu, Daniel Morton, Danny Driess, Daphne Chen, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dinesh Jayaraman, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Ethan Foster, Fangchen Liu, Federico Ceola, Fei Xia, Feiyu Zhao, Felipe Vieira Frujeri, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Gilbert Feng, Giulio Schiavi, Glen Berseth, Gregory Kahn, Guangwen Yang, Guanzhi Wang, Hao Su, Hao-Shu Fang, Haochen Shi, Henghui Bao, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homanga Bharadhwaj, Homer Walke, Hongjie Fang, Huy Ha, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jad Abou-Chakra, Jaehyung Kim, Jaimyn Drake, Jan Peters, Jan Schneider, Jasmine Hsu, Jay Vakil, Jeannette Bohg, Jeffrey Bingham, Jeffrey Wu, Jensen Gao, Jiaheng Hu, Jiajun Wu, Jialin Wu, Jiankai Sun, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jimmy Wu, Jingpei Lu, Jingyun Yang, Jitendra Malik, João Silvério, Joey Hejna, Jonathan Booher, Jonathan Tompson, Jonathan Yang, Jordi Salvador, Joseph J. Lim, Junhyek Han, Kaiyuan Wang, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Black, Kevin Lin, Kevin Zhang, Kiana Ehsani, Kiran Lekkala, Kirsty Ellis, Krishan Rana, Krishnan Srinivasan, Kuan Fang, Kunal Pratap Singh, Kuo-Hao Zeng, Kyle Hatch, Kyle Hsu, Laurent Itti, Lawrence Yunliang Chen, Lerrel Pinto, Li Fei-Fei, Liam Tan, Linxi "Jim" Fan, Lionel Ott, Lisa Lee, Luca Weihs, Magnum Chen, Marion Lepert, Marius Memmel, Masayoshi Tomizuka, Masha Itkina, Mateo Guaman Castro, Max Spero, Maximilian Du, Michael Ahn, Michael C. Yip, Mingtong Zhang, Mingyu Ding, Minho Heo, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Muhammad Zubair Irshad, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Ning Liu, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Osbert Bastani, Pannag R Sanketi, Patrick "Tree" Miller, Patrick Yin, Paul Wohlhart, Peng Xu, Peter David Fagan, Peter Mitrano, Pierre Sermanet, Pieter Abbeel, Priya Sundaresan, Qiuyu Chen, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Martín-Martín, Rohan Baijal, Rosario Scalise, Rose Hendrix, Roy Lin, Runjia Qian, Ruohan Zhang, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Shan Lin, Sherry Moore, Shikhar Bahl, Shivin Dass, Shubham Sonawani, Shubham Tulsiani, Shuran Song, Sichun Xu, Siddhant Haldar, Siddharth Karamcheti, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Subramanian Ramamoorthy, Sudeep Dasari, Suneel Belkhale, Sungjae Park, Suraj Nair, Suvir Mirchandani, Takayuki Osa, Tanmay Gupta, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Thomas Kollar, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Trinity Chung, Vidhi Jain, Vikash Kumar, Vincent Vanhoucke, Vitor Guizilini, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiangyu Chen, Xiaolong

Wang, Xinghao Zhu, Xinyang Geng, Xiyuan Liu, Xu Liangwei, Xuanlin Li, Yansong Pang, Yao Lu, Yecheng Jason Ma, Yejin Kim, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Yilin Wu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yongqiang Dou, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yue Cao, Yueh-Hua Wu, Yujin Tang, Yuke Zhu, Yunchu Zhang, Yunfan Jiang, Yunshuang Li, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zehan Ma, Zhuo Xu, Zichen Jeff Cui, Zichen Zhang, Zipeng Fu, and Zipeng Lin. Open X-Embodiment: Robotic learning datasets and RT-X models, 2025. URL https://arxiv.org/abs/2310.08864.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, et al. DINOv2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. URL https://arxiv.org/abs/2304.07193.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

Krishan Rana, Jesse Haviland, Sourav Garg, Jad Abou-Chakra, Ian Reid, and Niko Suenderhauf. SayPlan: Grounding large language models using 3D scene graphs for scalable robot task planning, 2023. URL https://arxiv.org/abs/2307.06135.

Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, Adrian Li-Bell, Danny Driess, Lachy Groom, Sergey Levine, and Chelsea Finn. Hi Robot: Open-ended instruction following with hierarchical vision-language-action models, 2025. URL https://arxiv.org/abs/2502.19417.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. ProgPrompt: Generating situated robot task plans using large language models, 2022. URL https://arxiv.org/abs/2209.11302.

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning, 2025. URL https://arxiv.org/abs/2407.08693.

Wentao Zhao, Jiaming Chen, Ziyu Meng, Donghui Mao, Ran Song, and Wei Zhang. VLMPC: Vision-language model predictive control for robotic manipulation, 2024. URL https://arxiv.org/abs/2407.09829.

APPENDIX

A EFFECT REQUESTS AND EXECUTION-TIME INTERVENTION

This section expands the Agent–VLA interface in Section 3.3. The interface connects three decisions at a common temporal scale: identifying the physical change needed now, evaluating actions against that requirement, and updating the requirement after observing the outcome.

A.1 VISUAL CONTEXT AND REQUEST GENERATION

The high-level agent is a fine-tuned Qwen3-VL-8B-Instruct model. Its input combines the task instruction, global and wrist-camera images from the current and preceding decision rounds, and the corresponding end-effector pose and gripper state. Images are ordered by time and viewpoint. At the first round, a history-availability indicator identifies the start of the episode. The two viewpoints describe the object–target arrangement and the local interaction around the gripper; the temporal pair provides evidence of changes in object position and contact during execution.

The agent produces a short structured response containing one effect request. The output vocabulary maps `approach_object`, `lift_object`, `transport_to_target`, and `place_at_target` to the four labels *approach*, *lift*, *transport*, and *place* used throughout the paper. A fixed instruction specifies their meanings: approach brings the gripper into an actionable position; lift raises the object; transport moves the held object toward the target; and place establishes the target spatial relation. Deterministic decoding provides the request used at each decision point.

The agent's temporal, multiview context supports request generation. The effect ranker uses the current image and robot state, together with the generated request and candidate action, as defined in Section 3.3. This division gives the agent a task-progress representation and the ranker an action-specific prediction problem. The invoked language instruction continues to condition the frozen VLA across successive requests.

A.2 LEARNING TO GENERATE EFFECT REQUESTS

The agent is adapted through task-conditioned supervised fine-tuning with LoRA. A training sample pairs the task and decision-time visual context with the effect required at that decision point. In simulation, geometric relations provide offline request annotations based on proximity to the object, object elevation, proximity to the target, and the target relation. Samples are drawn from trajectories with different action-selection behaviors and cover both forward progression and states requiring a renewed approach or grasp. Sampling across tasks and stages exposes the model to the same effect requirement under different scene configurations.

Supervised learning applies the language-model loss to the answer tokens encoding the request. Training and validation data are separated by initial state; evaluation uses held-out initial states. The visual input teaches the agent to infer the appropriate request from observations, while geometric annotations supply its training targets. This request supervision concerns the state before action selection. The ranker's supervision, detailed in Appendix B, concerns the transition produced by each proposed action.

For real-world adaptation, synchronized global and wrist views, end-effector poses, and gripper states form the same decision-time input. Human annotations specify the required effect from the task and these observations. The simulation-adapted agent is further adapted using the real-world request annotations, with simulation replay maintaining coverage of the shared effect vocabulary. Data are separated by recording scene or collection session to preserve trajectory-level separation.

A.3 EXECUTION PROCEDURE

The following procedure expands Equation 4. The symbols t, o_t, ℓ, r_t, K, H, a_t^k, and s_t^k retain their definitions from Section 3: decision round, current observation, subtask instruction, effect request, candidate count, chunk length, candidate action, and candidate score, respectively.

1. Observe the current scene and robot state, and assemble the agent's current and preceding visual context.
2. Generate r_t from that context and the active instruction ℓ.
3. Sample K candidate chunks from the frozen VLA using o_t and ℓ. In the simulation experiments, $K = 4$ and $H = 5$.
4. For a valid effect request, compute s_t^k for every candidate k and select the highest-scoring chunk. When the agent returns an abstention or an unparseable response, use uniform candidate selection for that round.
5. Execute the selected chunk, acquire the next observation, and update the agent's temporal context. Repeat while the subtask remains active and the control-step budget remains available.

For every valid request, selection compares the complete candidate pool by score. The maximum-score rule applies both when several proposals receive high scores and when all receive low scores; equal maximum scores are resolved by candidate order. After execution, the observed transition supplies the realized-effect label used for evaluation. Thus, action selection uses predicted compatibility, and effect evaluation uses the resulting physical transition. At the next round, the updated observation lets the agent revise its request and intervene again within the same subtask.

B Effect Annotation and Ranker Learning

B.1 Shared Physical-Effect Semantics

The contract uses the same effect labels for request generation, candidate supervision, and evaluation. Table 5 expands the annotation rule in Section 4.1. Each change is measured from the beginning to the end of the executed action chunk, using the object and target specified by the task.

Table 5: Physical conditions used to assign effect labels in LIBERO. Conditions are evaluated in the displayed priority order, and the first satisfied condition determines the label.

Effect	Condition on the action-induced transition
Place	The endpoint satisfies the task-defined target spatial relation.
Transport	Object–target distance decreases by at least $1\,\mathrm{cm}$.
Lift	Object height increases by at least $1\,\mathrm{cm}$, and endpoint end-effector–object distance is at most $8\,\mathrm{cm}$.
Approach	End-effector–object distance decreases by at least $1\,\mathrm{cm}$.

A transition may satisfy several physical conditions. The priority place, transport, lift, approach assigns one realized label to that transition, consistently across all methods. Transitions satisfying none of the four conditions receive labels outside the request vocabulary. Matching is then evaluated by the exact equality in Equation 1: the assigned label must equal the current request. This procedure makes the meaning of each reported Hit column precise, including transitions that combine vertical and target-directed motion.

B.2 Constructing Candidate Groups

Each simulation group fixes the observation, instruction, and request. The frozen policy generates candidates, each replayed from the same saved state. Transition labels become binary matching targets through Equation 5. Real-world groups likewise share starting conditions, with human annotations supplying realized-effect labels for ranker training.

This grouping connects differences in candidate controls to differences in realized effects. Positive and negative candidates can share the image and request while requiring different scores. Consequently, learning to distinguish them requires information about the proposed action and its relationship to the scene. The candidate-dependent query in Equation 3 provides this connection: robot state and request establish the decision context, and the candidate controls determine how the current visual patches contribute to its evaluation.

B.3 Complementary Roles of the Learning Objectives

Candidate-level classification and within-group ranking use the same labels at two complementary levels. The classification term increases the matching score of positive candidates and decreases that of negative candidates. Its positive-label weight accounts for the frequency of matching proposals in the training data. Every labeled candidate contributes, including candidates in uniformly positive or uniformly negative groups.

The ranking term compares each positive candidate with each negative candidate from the same group. Its penalty decreases as the positive candidate's score rises above the negative candidate's score. Because the comparison holds the scene and request fixed, it directly trains the ordering needed by maximum-score selection. Groups containing both labels provide these comparisons; groups containing a single label continue to contribute through classification. When a batch has no positive–negative pairs, its ranking term is zero, as defined in the main text.

Together, the objectives teach candidate-level effect recognition and request-specific ordering within a shared context. The frozen visual backbone supplies patch features, while the projection, query, and scoring components learn how those features relate to candidate actions. The action representation also enters the scoring function directly, retaining the proposed control sequence alongside the candidate-dependent visual summary. These roles motivate the visual-pooling and ranking-loss comparisons in Table 3.

C Evaluation Protocol and Ablation Definitions

C.1 Tasks and Comparison Structure

The nine simulation tasks in Table 1 span four kitchen scenes and two living-room scenes. They require placing black bowls, a white bowl, a moka pot, and a ketchup bottle at task-specific destinations, including plates, a drawer, cabinet tops, a stove, and trays. The language instruction identifies the manipulated object and destination; the online effect request specifies the physical change needed during the current stage of that task.

Each simulation method is evaluated for 70 episodes per task, giving 630 episodes over the nine tasks. The methods share the initial-state evaluation protocol, task goals, control-step budgets, and frozen action-policy checkpoint. Default executes the policy's default proposal. Random and Ours use four proposals of five control steps each: Random selects uniformly, and Ours scores candidates using the current request. Comparing Ours with Random evaluates request-conditioned selection at a common candidate budget; comparing Ours with Default measures the improvement over direct policy execution.

The real-world evaluation uses the AgileX Piper robot arm and four object-placement tasks. Each of the black pen holder, eggplant, corn cob, and banana is placed into a white plate. Thirty trials per object give 120 trials per method. The three methods retain their respective default, uniform, and effect-conditioned selection rules. The effect ranker is trained on real-world data for these tasks. Task success requires releasing the designated object and leaving it stably in the plate.

C.2 Aggregation of Success and Effect Matching

For task success, the numerator counts successful episodes and the denominator counts attempted episodes. The overall simulation rate combines all 630 episodes. Because every task has the same number of episodes, this is also the equal-weight mean of the unrounded task success rates. The real-world overall rate similarly combines the 120 trials for each method. Percentages are rounded to two decimal places after aggregation, and success-rate differences are computed from the underlying counts.

For effect matching, each executed chunk with a valid request is associated with its decision-time request and its realized transition label. For a given method and request, Hit pools the corresponding chunks across all nine simulation tasks and divides the number of matching chunks by the number carrying that request. Rounds in which the agent abstains or produces an unparseable request are excluded from the Hit denominator. The current request remains the reference label for evaluation under Default and Random, while their action-choice rules use the default proposal and uniform

selection, respectively. Thus, all three methods are evaluated for agreement with the agent's current requirement.

The two metrics characterize different parts of the execution loop. Hit measures whether the chosen chunk produces the requested local change. SR measures whether the resulting sequence reaches the task goal. Reporting them together connects the action-level intervention in the method to the episode-level outcome in the experiments.

C.3 Definitions of the Ranker Variants

All ranker variants retain the same frozen VLA, four-candidate pool, five-step chunk length, task set, and effect-label definitions. The changes below specify the component examined by each comparison in Table 3.

Without effect request. The request contribution is removed from both the candidate query and the scoring context. The image, robot state, candidate action, and request-matching training targets are retained. This comparison measures the contribution of supplying the current requested effect to candidate evaluation.

Global visual pooling. The mean of the projected visual patch vectors forms a common visual summary for every candidate in a group. Robot state, request, and candidate-action inputs remain available to the scoring function. This comparison measures the contribution of selecting spatial evidence according to the proposed action.

Without ranking loss. Training retains the weighted candidate-level classification objective and removes the within-group pairwise term. The architecture and online maximum-score rule are preserved. This comparison measures the contribution of explicitly learning the ordering between matching and nonmatching candidates from the same state.

Without robot state. The robot-state contribution is removed from the candidate query and scoring context. The image, effect request, candidate action, and both training objectives are retained. This comparison measures the contribution of robot-state information to predicting an action's physical effect. Default is reported separately as the direct-execution reference in the same results table.

D Supplementary Result Analysis and Execution Visualization

D.1 Distribution of Task-Level Gains

The per-task rates in Table 1, with 70 episodes per task, correspond to 265 successful episodes for Default, 275 for Random, and 323 for Ours. Ours therefore completes 58 more episodes than Default and 48 more than Random over the 630-episode evaluation. Improvements over Random occur in all nine tasks, with per-task gains ranging from one to ten successful episodes. The upper-drawer task contributes ten additional completions, while the middle-bowl and living-room left-bowl tasks contribute seven and nine, respectively. This distribution shows that the overall gain extends across several object–destination configurations.

In the real-world results, the corresponding totals are 58, 56, and 70 successes for Default, Random, and Ours. Relative to Random, Ours adds three successful trials for the pen holder, three for the eggplant, five for the corn cob, and three for the banana. These object-level differences account for all 14 additional completions in the 120-trial evaluation and show gains across the four manipulated objects.

D.2 Effect Matching and Component Contributions

The stage-wise results connect these task outcomes to the immediate physical effects targeted by the interface. Relative to Random, Ours improves Hit by 9.30, 8.11, 11.94, and 8.67 percentage points for approach, lift, transport, and place. Improvements cover reaching the object, raising it, moving it toward the destination, and establishing the target relation. Together with the task-level results, this pattern supports using effect requests throughout the execution of a manipulation subtask.

The ablations further identify which parts of the ranker contribute to this pattern. Removing the request produces the largest SR decrease, 10.48 percentage points, together with decreases in all four Hit columns. Global pooling reduces SR by 8.73 points, linking candidate-dependent scene evidence to the success of action selection. Removing ranking supervision reduces SR by 7.62 points, supporting the use of within-state candidate comparisons. The robot-state ablation completes 299 of 630 episodes, yielding 47.46% SR; the full model completes 24 additional episodes, a 3.81-point gain. Across these comparisons, local matching and task completion favor the complete ranker.

D.3 Physical Interpretation of the Effect Vocabulary

Figure 3 uses selected frames from four recorded real-world manipulation sequences to illustrate the physical progression described by the four effect labels. Within each sequence, the object and plate remain the task reference, while the relevant physical change evolves with the gripper–object configuration. This progression makes the interface's temporal role concrete: one task instruction persists as successive stages call for different effects.

Execution sequences across objects. The rows of Figure 3 show black pen holder, banana, corn cob, and eggplant placement. Each row follows one object from gripper approach through lifting and transport to placement in the white plate. The shared stage layout relates the effect vocabulary to changes in object position and gripper interaction across the four tasks, providing visual context for the four-object evaluation in Table 4.

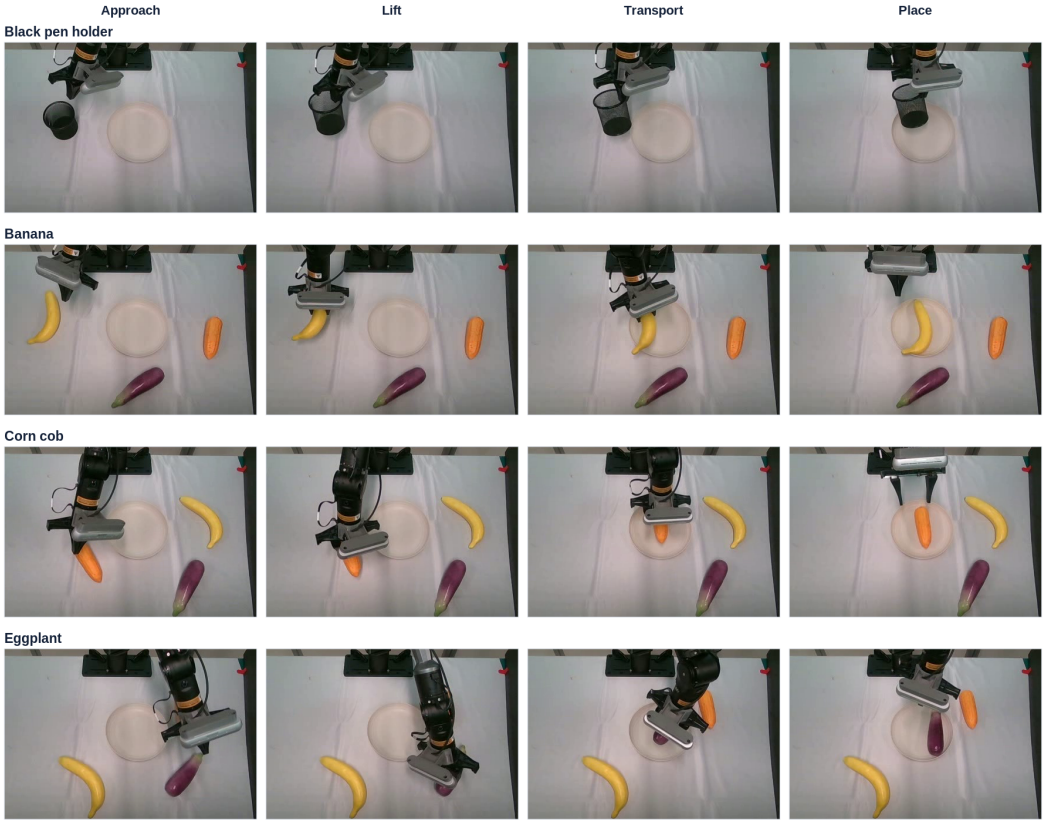


Figure 3: Recorded real-world placement sequences: black pen holder, banana, corn cob, and eggplant, from top to bottom. Columns show approach, lift, transport, and placement keyframes. All four tasks use the same global-view crop dimensions and a shared stage layout.