ATLAS: Agentic Harness for Whole-Body Intelligence from Human Video

Junli Ren[*1,3], Kaixiang Yao[*1], Junfeng Long[*1], Tao Huang[3], Huayi Wang[3], Yuxiao Chen[2], Wenhao Ding[2], Yan Wang[2], Wenjie Luo[2], Boris Ivanovic[2†], Masayoshi Tomizuka[1†], Ping Luo[3†], Pieter Abbeel[1†], Marco Pavone[2,4†], Yuxin Chen[1‡], Ran "Thomas" Tian[1,2‡]

[1]UC Berkeley [2]Autonomous System & Physical AI Research Group, NVIDIA [3]The University of Hong Kong [4]Stanford University

Project page: atlas-humanoid.github.io

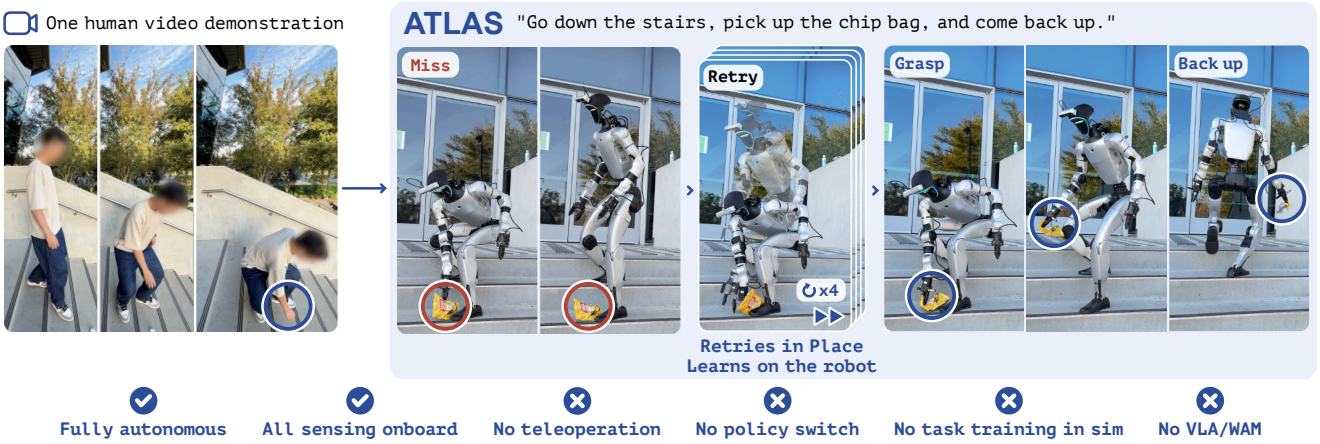


Figure 1. ATLAS in the real world. From one human video demonstration, with no robot demonstrations, the robot goes down an outdoor staircase and comes back up with the chip bag its prompt names. Its first grasp misses, so it retries in place and grasps the bag, learning on the robot as it goes. No pretrained vision-language or vision-language-action model is fine-tuned.

ABSTRACT

Scaling humanoid whole-body intelligence is often pursued through large-scale training of vision-language-action (VLA) or world action models (WAMs). Yet recent advances in pretrained vision-language agents provide reusable knowledge about objects, task goals, and physical interactions. This motivates acquiring new whole-body skills by adapting human motion references to new tasks via pretrained task reasoning and learning execution corrections online, **without** large-scale VLA or WAM training. We present ATLAS (Agentic Test-time Learning and Adaptation System), an agentic harness that edits human motion for humanoid loco-manipulation from one human video demonstration and one prompt. The harness splits the motion into navigation and interaction phases as specified by a pretrained agent, plans the navigation, and aligns each interaction phase with the target object by its affordance and perceived geometry; a frozen whole-body tracker then follows the edited reference. During execution, online model-based RL mitigates the tracking gap between the edited reference and the robot's motion, while a slow system lets the agent refine the harness's grounding of each scene from robot observations. Success checks detect failures and trigger retries in place, generating new execution data and observations for further control and grounding adaptation. To the best of our knowledge, ATLAS is the first agentic whole-body intelligence system to achieve this without task-specific robot demonstrations, VLA or WAM training, or task-specific simulation training. We demonstrate ATLAS on a real humanoid performing coordinated whole-body loco-manipulation, including tasks that combine stair traversal with object manipulation, using only onboard sensing. In both simulation and real-world experiments, ATLAS acquires skills from one human video demonstration and one prompt per task and transfers them across objects and scenes. On

[*]Co-first authors. †Equal advising. ‡Project leads. The work was done at UC Berkeley, in academic collaboration with NVIDIA.

real-robot floor picks of three objects, online correction raises success from 25% to 63%. On three whole-body pick tasks of the SIMPLE benchmark, a single human video serves all three tasks, and ATLAS succeeds in 29, 28 and 30 of 30 episodes with onboard sensing, competitive with VLA, WAM and imitation policies fine-tuned on 100 robot demonstrations per task.

1 Introduction

Scaling humanoid whole-body intelligence with vision-language-action (VLA) or world action models (WAMs) typically requires substantial robot or human data and compute to map visual observations and task instructions to robot actions [33, 47]. Yet pretrained vision-language agents increasingly provide reusable knowledge about objects, task semantics, and physical interactions [20, 23]. They can turn this knowledge into robot programs [29], geometric constraints [23], or commands through closed-loop action interfaces [20]. Recent agentic tabletop manipulation systems need no task-specific robot demonstrations and have outperformed several SOTA VLA/WAM baselines on manipulation benchmarks [12, 48]. For humanoids, this motivates enabling pretrained vision-language agents to perform new tasks requiring coordinated whole-body motion, without further large-scale VLA/WAM training or task-specific simulation training.

Human video can provide a whole-body reference for an agent's plan through motion recovery and retargeting [4, 38]. The reference must still be grounded in the instructed objects and scene and adapted to the robot's embodiment. Our key idea is to let a pretrained agent specify reference edits while the robot learns from its attempts to refine grounding and correct the frozen tracker's end-effector commands. For example, to reuse a banana-picking motion to collect a paper ball, the robot must locate the ball and align the reach with its geometry. Even with the target correctly placed, tracking error can leave the hand short of contact.

We present ATLAS (Agentic Test-time Learning and Adaptation System), an agentic harness for humanoid loco-manipulation from one human video demonstration and one prompt. ATLAS connects a vision-language agent (System 2) to a frozen whole-body tracker (System 0) only through edited motion references, replacing an intermediate policy pretrained on robot data. Once per task, the agent segments the demonstration into steps and specifies target objects, contact affordances, and success checks. On each attempt, the harness grounds these affordances in perceived geometry, navigates into position if needed, and aligns the reference's reach with the contact poses (Sec. 3 and Fig. 3). A progress indicator then applies the success checks, and a miss is retried in place.



Figure 2. One human video demonstration transfers to new objects and scenes. A person picks a cup off the floor. On real hardware, the same motion drives the robot to grasp a mango, a paper ball and a bottle in three scenes. The agent writes each task's specification, and no pretrained vision-language or vision-language-action model is fine-tuned.

The harness uses execution feedback to adapt both grounding and control. During execution, its fast system uses online model-based RL to learn the tracker's command-response dynamics from scratch in each session and corrects the end-effector command at every control step during reaching (Sec. 3.4). Outside the control loop, its slow system uses the same agent to revise the perception phrases and the progress indicator from robot observations, retaining only revisions that improve performance when replayed on saved observations (Sec. 3.5). Retries supply further experience for both systems.

We demonstrate ATLAS on a real humanoid with onboard sensing, including stair traversal with manipulation (Figs. 1 and 5) and motion reuse across objects and scenes (Fig. 2). At a fixed apple placement in simulation, our model-based learner achieves about 63% lower median tracking error during the pre-grasp hold than a model-free actor. Both correct the end-effector command at every control step. Across real-robot configurations with differing calibrations, our learner achieves 63% floor-pick success versus 25% without correction (Sec. 4.6). In held-out replay of robot observations, agent review improves object recall and reduces false success judgments (Sec. 4.4). On the public SIMPLE benchmark, ATLAS reuses one human video across three loco-manipulation tasks without task-specific robot demonstrations, achieving success rates competitive with VLA, world-action-model, and imitation policies fine-tuned on 100 robot demonstrations per task (Tab. 1).

2 Related Work

VLA and WAM for humanoid whole-body intelligence. Hierarchical stacks such as Helix 02 [17] and Dyna-2.1 [15] combine vision-language reasoning (System 2), learned action generation (System 1), and whole-body control (System 0). Training VLAs [33, 37] and WAMs [47] to map visual observations and instructions to actions requires substantial data and compute.Pretrained agents reuse task knowledge for reasoning and grounding [20, 23, 29]; manipulation results without task-specific robot demonstrations [12, 48] suggest a route to new behaviors without additional task-specific VLA/WAM training.

Whole-body controllers. Pretrained whole-body controllers provide reusable motor capabilities from motion references or goal commands [28, 32]. However, embodiment differences and mismatches between training and deployment references can limit tracking accuracy [31]. For loco-manipulation, residual end-effector errors can prevent the robot from reaching the contact targets required for successful interaction. Keeping ScaleBFM [49] frozen, ATLAS uses online model-based RL to learn its command-response dynamics from robot attempts and train an actor that corrects end-effector commands during reaching.

Agents and harnesses for humanoids. Vision-language agents are increasingly used to bridge task reasoning and humanoid control, offering an alternative to VLA-based intermediate policies. HomeBody [25] and KINO [9] use agents to select and sequence reusable skills or intermediate whole-body pose goals, leaving motion generation and execution to downstream planning and control modules. KPI [42] uses an agent's parameterized wrist references and contact objectives to adapt arm contact behavior online. HuGo [11] generates closed-loop policy code and refines it through task-specific simulation rollouts, with further refinement from real-robot feedback. A remaining challenge is to enable new whole-body task behaviors without handcrafting task-specific skills or requiring task-specific simulation optimization. ATLAS uses pretrained agent reasoning to adapt human video references for a frozen whole-body tracker, refining grounding and execution online without task-specific robot demonstrations or additional VLA/WAM training.

From human video to humanoid motion. Human videos provide motion and interaction structure for acquiring new humanoid behaviors. VideoMimic reconstructs human motion and scene geometry from video, then learns contextual control policies in simulation [1]. Other methods adapt video-derived references through object-aware retargeting [27], closed-loop posture and contact adjustment [21], or contact-aware trajectory optimization [5]. RoboReact further uses a VLM to refine video-derived skill keyframes and interaction parameters from physical calibration rollouts [22]. ATLAS builds on reference-based execution by coupling agent-driven grounding refinement for the instructed objects and scene with online model-based RL that corrects end-effector commands within each reach.

On-device learning. Residual learning augments existing robot controllers and policies with learned corrections [3, 16, 26, 50]. For frozen whole-body controllers, adaptation can operate on motion commands through simulation-based optimization [46] or offline-learned kinematic and odometry models [14]. OSRAM fine-tunes a simulation-meta-trained command-response model on hardware and uses MPPI to optimize command sequences online [24]. At this command interface, ATLAS learns a command-response model and a correction actor from scratch in each deployment session. Unlike OSRAM, its model needs no meta-training in simulation, and a trained actor replaces online sampling-based optimization; unlike iterative learning control [6], which updates a correction between repetitions, it also corrects the command within each reach. It fits the model to robot attempts and trains the actor through model rollouts [13, 30, 45], yielding end-effector command corrections at every control step during reaching.

3 ATLAS: Agentic Test-time Learning and Adaptation System

3.1 Task setup and deployment loop

A related human video already contains the whole-body coordination a task needs; what changes across tasks is mainly the interaction with the object. ATLAS therefore reuses the demonstrated motion and adapts it to the instructed interaction, even when the objects differ from those in the video (Fig. 3). On each attempt, the harness locates the instructed object, walks the robot to it if needed, and shifts the demonstrated end-effector target onto it; the frozen tracker executes the edited motion, and a progress indicator checks the end state and retries failures in place. Two kinds of error remain, and two learners reduce them from the robot's own attempts: the fast system corrects the tracker's end-effector error during execution (Sec. 3.4), and the slow system lets the agent revise what perception looks for and how success is judged (Sec. 3.5). An *attempt* ends in success or failure, an *episode* groups the

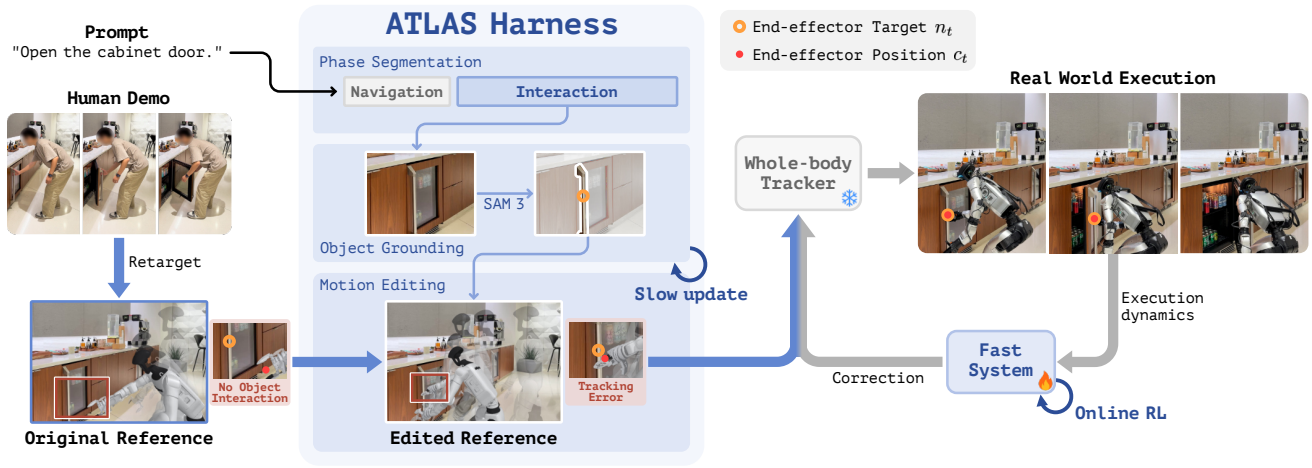


Figure 3. ATLAS links a vision-language agent (System 2) and a frozen whole-body tracker (System 0) through edited motion references. Left: human video supplies whole-body motion; the agent specifies interaction steps, objects, contacts, and success checks. Right: onboard perception grounds contact targets, and the harness aligns the reference for execution. The fast system learns end-effector command corrections; the slow system revises what perception is asked to find and the success checks through replay validation. Here, a human cabinet-opening video and the prompt "Open the cabinet door." drive the robot to open a cabinet.

attempts at one object placement, and a *session* keeps experience across episodes.

3.2 From human video to grounded motion

Motion reference. GVHMR [38] recovers the person's motion and GMR [4] retargets it to the robot, giving a reusable reference of body poses and end-effector trajectories. The agent splits it into interaction steps, such as a pick and a place, and marks each step's interaction start, contact, settle, and end frames.

Task specification. The agent specifies each step only in text and a few parameters, which the slow system can later revise. From the instruction, measurements of the reference such as the contact height and action axes, and a description of the robot's end effector, it writes an object phrase (and a destination phrase if needed), contact and closure parameters, and a visible end state that the progress indicator checks for success. For example, when a banana pickup is reused to pick up a paper ball with a fingered hand, it names the paper ball, chooses an `enclose` closure, and defines success as the ball held above the floor.

Grounding and navigation. At run time, SAM 3 [7] segments the phrase's *grounding target*, the object or the part to interact with, such as a handle, and depth gives its center, extent, and axes, providing real-time, closed-loop feedback on the object position. From this geometry and the specified interaction, the harness computes the wrist orientation and the *target contact point* $p^\star$, for example the center of a handle. Navigation ensures that the interaction starts from the same pose relative to the object as in the retargeted reference: the *interaction station* is derived backward from the reference's interaction, as its pelvis pose at the interaction start relative to its contact point, and a whole-body walking planner [41] generates walking references that the frozen tracker follows to it.

3.3 Editing the reference and executing it

The harness changes only the interacting end effector's target and, for higher objects, the body height, so the rest of the motion keeps the demonstrated whole-body coordination that the frozen tracker can follow.

Placing the reference. At the *handoff*, the robot stops walking and starts the demonstrated motion from where it stands: the reference is turned and shifted on the floor so that its starting pelvis coincides with the robot's. To compare targets, measurements, and corrections while the body moves, all positions are then expressed in the *attempt frame*, fixed to the floor at handoff and pointing forward, left, and up; the robot knows where its pelvis is in this frame from leg kinematics, without external localization. The *contact point* is the part of the end effector meant to touch the object, at a fixed offset from the wrist.

Editing the end-effector target. Placed this way, the reference still moves the end effector toward where the person's object was, so the harness shifts the end-effector target onto $p^\star$, gradually so that the reach stays smooth. With τ the annotated contact frame and c_t^{ref} the placed reference's contact point at control step t, the target is

$$n_t = c_t^{\mathrm{ref}} + \alpha_t\left(p^\star - c_\tau^{\mathrm{ref}}\right), \tag{1}$$

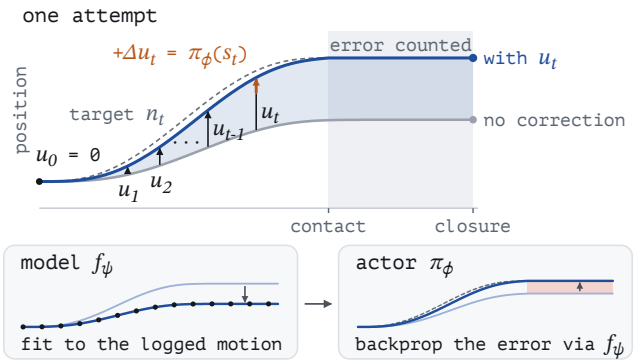


Figure 4. The fast system accumulates bounded command corrections to reduce contact error during the hold. In this schematic, the uncorrected tracker undershoots n_t; the shaded band shows the shift u_t produces. Logged motion trains the dynamics model f_ψ, and backpropagating hold error through its rollouts trains the actor π_ϕ.

where α_t rises from 0 at *reach onset*, the start of the wrist's final approach to the object, to 1 at τ in proportion to the wrist path traveled, so that $n_\tau = p^\star$. New head-camera estimates update $p^\star$ during the crouch, closing the loop on the object position.

Adapting height. For higher objects, body and arm references rise by the height difference over the reach, with the feet on the floor and the pelvis capped at standing height; for lower objects, only Eq. (1) lowers the end effector.

Whole-body tracking. One tracker, ScaleBFM [49], follows both the walking and the interaction references at 50 Hz without switching policies, and stays frozen during deployment; for stair traversal, we fine-tune it beforehand on general stair-climbing data, not on any task. For load-bearing grasps, the reference pauses at contact with the end effector open (the *hold*) before closing; pushes, pulls, and presses close at contact.

3.4 Fast system: command correction via online model-based RL

Even with correct grounding, the frozen tracker leaves systematic end-effector errors that vary across attempts: its response depends on where the command lies relative to the body and on the posture along the reference. The fast system learns how the robot responds to end-effector commands with online model-based RL and corrects the command from reach onset through the hold (Fig. 4).

What the actor controls. Let c_t be the contact point measured by forward kinematics. To bring c_t to n_t during the hold, the actor adds an offset u_t on attempt-frame axes to the interacting wrist's position command (the *harness command*); n_t and the tracker's weights stay unchanged. The actor outputs increments $\Delta u_t = \pi_\phi(s_t)$ of at most 5 mm per control step that accumulate,

$$u_t = \mathrm{clip}(u_{t-1} + \Delta u_t), \tag{2}$$

starting from $u = 0$ at reach onset; clipping keeps u_t within fixed bounds. Because the state s_t (defined below) includes u_{t-1}, the actor can still reach any correction within these bounds.

Learning how the robot responds. After each attempt, a dynamics model learns how the contact point responds to the commands, working relative to the shoulder. Writing $c_t = b_t + h_t$, where b_t is the interacting arm's recorded shoulder position, it predicts

$$\hat{h}_{t+1} = \hat{h}_t + f_\psi(z_t), \qquad \hat{c}_{t+1} = b_{t+1} + \hat{h}_{t+1}, \tag{3}$$

where z_t holds the reference phase (0 at reach onset, 1 at contact), h_t and its last change, and recent and look-ahead commanded contact points relative to b_t. Each command is the harness command plus the applied correction. We fit an ensemble of such models with randomized priors [36] and a multi-step prediction loss on all attempts of the session except falls, including failed grasps; rollouts start from a measurement and feed back the predicted $\hat{h}$.

Training the actor. The actor solves a finite-horizon RL problem with state s_t, action Δu_t, and transitions given by the robot under the frozen tracker. With the model fixed, it learns by backpropagating the predicted hold error through the model's reach-and-hold rollouts [13]. The actor sees what the model sees plus the target: its state s_t adds to z_t the errors $n_{t+k} - c_t$ for $k = 0, 5, 10$, the applied correction, and the correction not yet applied because of command delay. Each rollout starts from a recorded attempt's measured reach-onset state and replays its shoulder trajectory and harness command, substituting the actor's corrections under the execution limits; this treats body motion as fixed context, an approximation for bounded corrections around a recorded movement. For load-bearing grasps, the reward is the contact error during the hold until closure,

$$r_t = -\|n_t - \hat{c}_t\|^2, \tag{4}$$

on those axes; the undiscounted return sums it over the ensemble and replayed attempts, with a penalty on correction changes [30]. The reach itself has no error term: corrections during the reach count only through the hold accuracy they lead to. This kinematic reward needs neither task-success labels nor ground-truth object state.

Goal augmentation. Goal augmentation [2] reuses each attempt for other targets. Moving $p^\star$ by d shifts n_t and the harness command by $\alpha_t d$ (Eq. (1)); since

the model reads the harness command and the correction only through their sum, shifting the correction by $-\alpha_t d$ leaves the issued commands unchanged, so the attempt is also an exact sample for $p^\star + d$. Each replay draws d as another attempt's $p^\star$ minus its own, both relative to the shoulder.

Online execution and updates. The model and actor start from scratch for each task step in each session; during the first 5 attempts the robot explores around zero correction. Both train in the background after every attempt, and each attempt loads the newest actor that passes a validity check. After closure the correction ramps to zero, and the frozen tracker executes the rest of the motion, including sustained contact.

3.5 Slow system: refinement through agent review

The slow system fixes failures that a command correction cannot: perception misses the target or segments the wrong object, or the progress indicator accepts an empty grasp.

Proposing revisions. Outside the control loop, the agent reviews the instruction, saved camera frames, and attempt summaries. When the phrase is vague or wrong, or the instruction covers several objects, it proposes one specific phrase per object, grounded in turn as the robot lifts each (Fig. 8). When carry frames show an empty grasp that was judged successful, it revises how the indicator checks the end state.

Validating and applying revisions. A revision is kept only if it does better when replayed on saved observations: a phrase must raise object detections on held-out frames, net of wrong-object detections, and an indicator revision must reduce disagreement with hindsight labels from later frames without adding false successes. Accepted revisions take effect at the next attempt.

4 Experiments and Results

We evaluate whole-body skill acquisition from one human video and one instruction (Sec. 4.2) and on diverse hardware tasks (Sec. 4.3), the slow system (Sec. 4.4), the reuse of motion references across objects, scenes, and object heights (Sec. 4.5), and the contribution from the fast system (Sec. 4.6).

4.1 Setup

Robot and simulation. Real-robot experiments use a Unitree G1 humanoid with onboard RGB-D sensing and joint-state measurements. The frozen whole-body tracker and actor run on the robot, while an external PC performs vision-language inference, segmentation, and updates to the dynamics model and actor. We use Claude Opus 5.5 as the agent. Simulation experiments use Isaac Lab with the same learning and execution framework, and the SIMPLE benchmark.

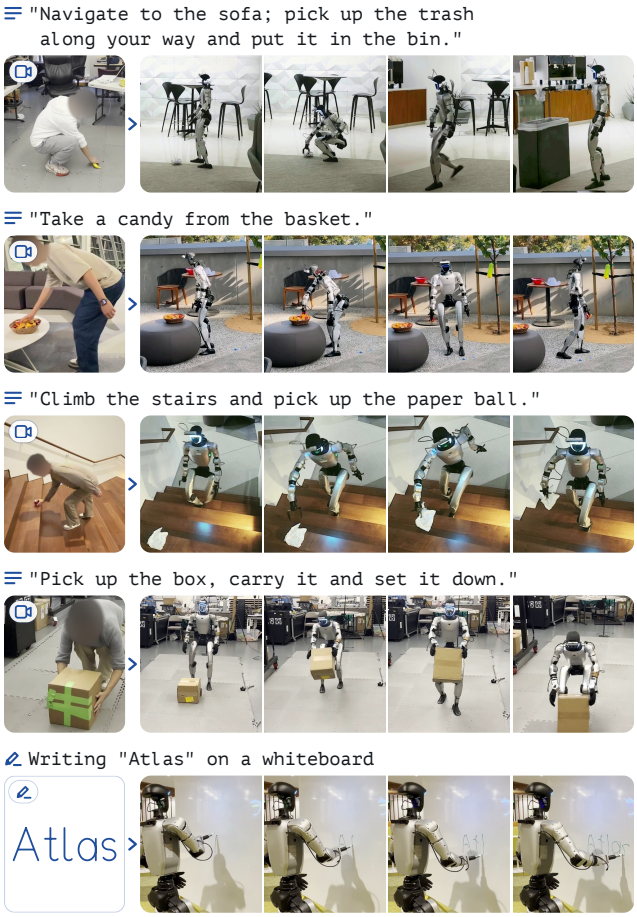


Figure 5. Tasks on the real robot. The first four rows each show the prompt, the human video demonstration (left) and the robot's run. In the last row the humanoid writes "Atlas" on a whiteboard, from the word's strokes (Sec. 4.3).

4.2 Acquire whole-body skill from human video

Human motion supports diverse whole-body interactions. On the real robot, ATLAS performs object pickup, box transport, and stair traversal with manipulation, each from one related human video and one instruction (Fig. 5).

One reference supports competitive performance across three tasks. With one human video and no task-specific robot demonstrations, ATLAS succeeds in 29/30, 28/30, and 30/30 episodes on *XMovePick*, *XMoveBendPick*, and *BendPick* of the public SIMPLE loco-manipulation benchmark [43], within one success of the best VLA, world-action-model, or imitation policy fine-tuned on 100 robot demonstrations per task (Tab. 1). For a fair comparison, ATLAS runs in SIMPLE's own environment, with its physics, head camera, reset states, and success check, in the same setting

Method	Training demos	XMovePick L0	XMovePick L1	XMovePick L2	XMovePick Total	XMoveBendPick L0	XMoveBendPick L1	XMoveBendPick L2	XMoveBendPick Total	BendPick L0	BendPick L1	BendPick L2	BendPick Total
Ψ_0 [44]	100 robot	10	10	6	26	10	9	9	28	10	10	10	**30**
GR00T N1.6 [33]	100 robot	10	10	7	27	4	4	1	9	7	7	6	20
$\pi_{0.5}$ [37]	100 robot	7	5	1	13	0	0	0	0	10	10	8	28
DreamZero [47]	100 robot	10	10	10	**30**	0	0	1	1	9	9	8	26
ACT [51]	100 robot	10	10	5	25	9	10	10	**29**	10	9	9	28
Diffusion Policy [10]	100 robot	3	3	2	8	0	0	0	0	10	8	6	24
ATLAS (ours)	1 human	10	10	9	29	9	9	10	28	10	10	10	**30**

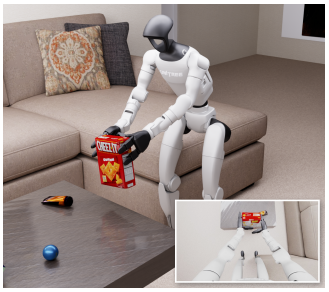


Table 1. On the public SIMPLE benchmark, one human video is competitive with policies fine-tuned on 100 robot demonstrations per task. Successes per level (/10) and overall (/30), using SIMPLE's success check. L0: distractors/materials; L1 adds extreme lighting; L2 adds box-pose perturbations. Baselines are fine-tuned on 100 robot demonstrations per task; scores from SIMPLE [43]. ATLAS reuses one human video across all tasks with onboard sensing and no task-specific robot demonstrations. Right: SIMPLE render of XMoveBendPick with a simultaneous head-camera inset. **Bold**: best total.

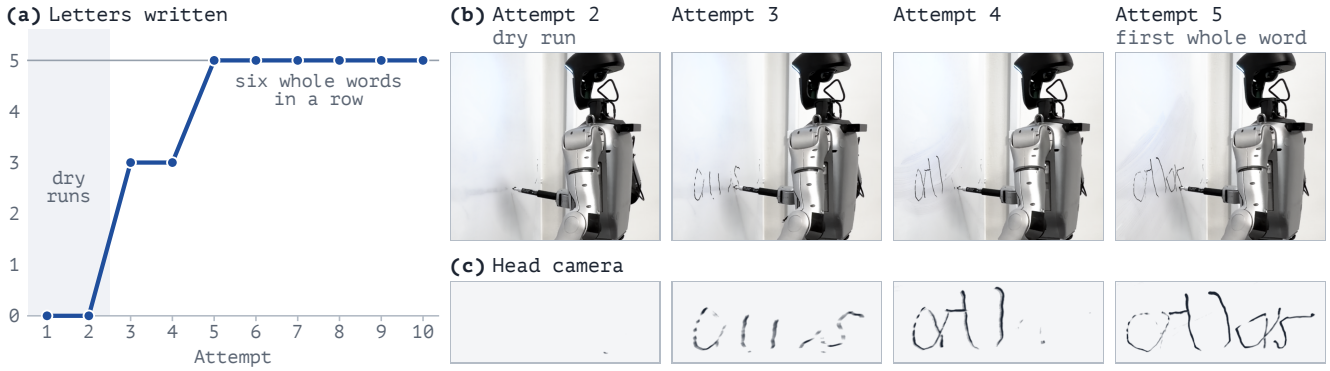


Figure 6. The humanoid writes the whole word six times in a row. (a) Letters written in each attempt of the take, out of five; a letter counts as written when ink lies along more than half of its stroke length in the head camera's images of the board. (b) Attempts 2–5 at the end of the last stroke, from the last dry run to the first whole word; earlier marks are toned down. (c) The same attempts in the robot's head camera, after each attempt with the pen up; the board's shading is evened out.

Task	Success
(a) Pick up the trash on the floor	38/60
(b) Place the trash into the trash can	11/11
(c) Lift the box from the floor	17/20

Table 2. Success on the real-robot tasks of Fig. 7. (a) The attempts of our learner on the three objects of Tab. 3. (b) Trials that ended with the trash in the trash can, out of those in which the robot picked it up. (c) All three box sizes together.

as our other experiments. Its fast-system learners start from zero at the first episode; because box pickups tolerate centimeter-level end-effector errors, the edited reference already reaches suitable grasp points before any correction is learned, so these scores mainly reflect the harness.

4.3 Quantitative investigation on diverse hardware tasks

Writing on a whiteboard. With a writing harness and the fast system of ATLAS, the humanoid writes the whole word "atlas" six times in a row on a wall-mounted whiteboard, with a marker fixed to its right wrist (Fig. 6). In one uninterrupted ten-minute take, it makes ten consecutive attempts: two dry runs aimed short of the board, two partial words, and then six whole words, with ink along more than 90% of their stroke length on average (Fig. 6a). The marker leaves ink only while its tip touches the board, so the harness measures the board's plane with the head camera eight times per second and keeps the strokes on it while the standing body moves by up to 28 mm (Appendix A).

Putting trash into a can and lifting boxes. The full system picks objects off the floor in 38 of 60 attempts (Sec. 4.6), puts the trash it picked up into a trash can in all 11 trials, and lifts boxes of three sizes in 17 of 20 attempts (Fig. 7 and Tab. 2).

4.4 The slow system improves grounding and success assessment

Online review enables recovery from grounding failures. Agent review recovers a pickup that starts from a wrong object description: a simulated red-apple pickup given a green-apple description succeeds in 37/60 attempts with the slow system and 0/60 without it, including failed attempts. Each condition runs

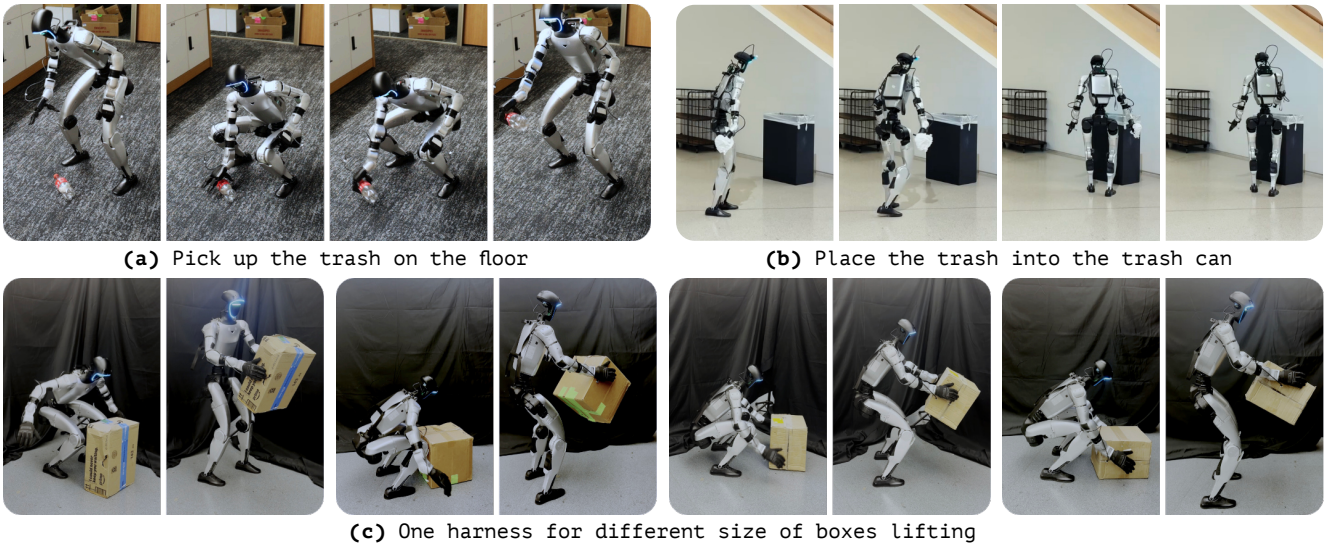


Figure 7. The full system completes diverse real-robot tasks. The humanoid (a) picks trash off the floor, (b) puts it into a trash can, and (c) crouches to lift boxes of three sizes with one harness, the smallest both standing and lying. Success rates are reported in Tab. 2.

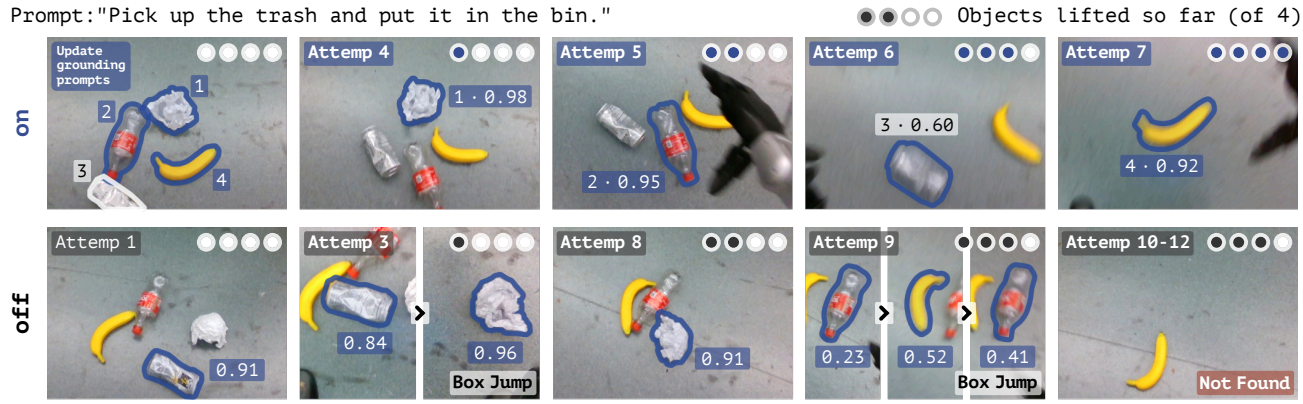


Figure 8. The slow system replaces the vague phrase "the trash." with one phrase per object, written by the agent from the robot's own views. Same scene and prompt in both runs; tabs give the attempt and the time since the first look, filled dots the objects lifted so far, and box labels the phrase number and the SAM 3 score. Top: the agent writes one phrase per object; phrase 3 lands on object 1 and is dropped, and the can is found with phrase 2 once the bottle is gone. Bottom: every attempt asks for "the trash.", the box jumps between objects, and the banana is never found.

three sessions of 20 attempts with the fast system enabled.

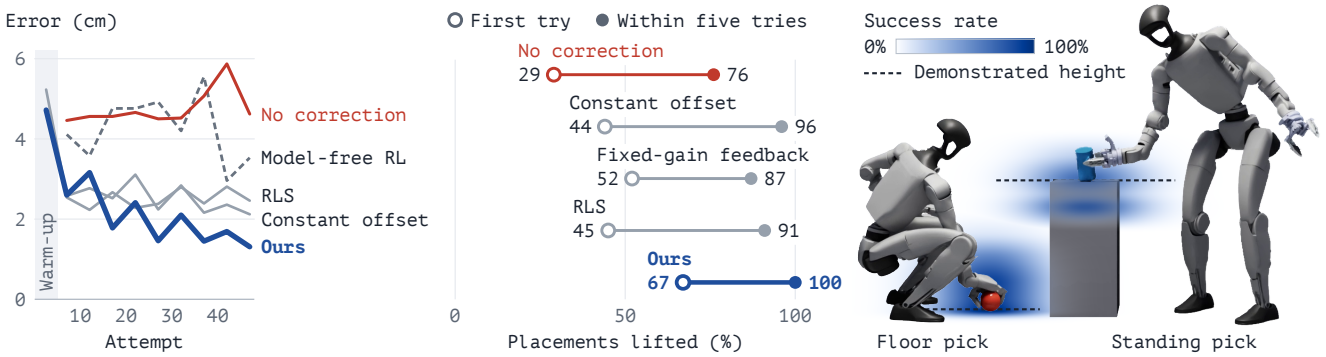
Revisions improve perception and success assessment. Revised phrases raise recall on held-out frames of real-robot runs from 18.9% to 78.3%, above a segmenter threshold tuned on the same early frames (66.1%, Holm $p = 0.049$). The candidates are proposed and selected on early frames of four runs where the initial phrase failed. Revised success checks reduce disagreements with hindsight labels from 39 to 24 across 103 held-out real-robot attempts: false successes on empty grasps fall from 42% to 14%, while false misses on held objects rise from 29% to 45%.

The robot collects multiple objects from one instruction. With the slow system, the robot clears all four objects of a scene from the instruction to pick up the trash, in 7 attempts and 2 min 23 s; without it, the robot lifts 3 objects in 12 attempts and never finds the banana (Fig. 8). Both runs start from the phrase "the trash.".

4.5 Reuse motion references across objects, scenes, and heights

Reuse across objects and scenes. On the real robot, the same banana-picking reference grasps a banana, a tennis ball, and a paper ball in different scenes (Fig. 2).

Adaptation to object height. One demonstration covers a band of object heights around the demonstrated



(a) Error at one placement (b) Retry in place (c) Reach of one demonstration

Figure 9. Our learner keeps lowering the error at one placement, retries recover most first-try misses, and one demonstration covers a band of object heights. (a) Simulated apple pick at one placement: median end-effector error over the hold in bins of five attempts, pooled over three sessions per learner. (b) Real robot: share of placements lifted at the first try (hollow) and within five tries (filled), pooled over the objects of Tab. 3. (c) Simulation, with the object raised on a support: each robot is shown at its demonstrated height, and the glow's shade is the success rate over ten attempts at each height, interpolated between heights.

Learner	Banana		Can		Apple	
	Success (%) ↑	Error (cm) ↓	Success (%) ↑	Error (cm) ↓	Success (%) ↑	Error (cm) ↓
No correction (control)	15.0 (3/20)	9.5 / 8.7	35.0 (7/20)	8.4 / 8.0	25.0 (5/20)	8.6 / 8.4
Constant offset	45.0 (9/20)	4.8 / 4.2	55.0 (11/20)	7.1 / 6.5	45.0 (9/20)	4.5 / 3.6
Fixed-gain feedback	50.0 (10/20)	**2.5** / **1.9**	35.0 (7/20)	5.5 / 4.7	35.0 (7/20)	3.0 / 2.8
Recursive least squares	45.0 (9/20)	3.5 / 3.2	50.0 (10/20)	4.8 / 3.1	20.0 (4/20)	3.5 / 2.9
Online model-based RL (ours)	**55.0** (11/20)	3.2 / 3.2	**80.0** (16/20)	**2.9** / **2.7**	**55.0** (11/20)	**2.6** / **2.4**

Table 3. Our learner achieves the highest observed success on all three real-robot pickups. The operator freely places floor objects in front of the robot, which picks with its right end effector and retries misses in place. We score 20 attempts per method and object: the control's first 20; learners' first 20 after 5 warm-up attempts, except can fixed-gain feedback (last 20). Success requires standing up holding the object; parentheses give successes/attempts. Error: mean / median distance from the end-effector contact point to its target at contact. Sessions differ in contact and target calibration; our apple session also uses a camera correction (Sec. 4.6). Bold: best among learners; underlined: second.

one (Fig. 9c). The floor-pick reference, demonstrated at 0 cm, lifts the apple in 10, 9, 9, 6, and 1 of ten attempts at 0, 5, 10, 20, and 30 cm and never at 40–80 cm; the standing reference, demonstrated at a 57 cm grasp height on a 51 cm support, lifts the can in 8, 5, 7, and 2 of ten at 40, 45, 50, and 60 cm and never at 20, 30, 70, or 80 cm. In simulation, we vary support height and reuse the reference and task specification with the height adjustment of Sec. 3.3; each height starts with a fresh learner and 5 warm-up attempts before 10 scored trials.

4.6 The fast system improves execution accuracy and grasp success

Baselines. We compare online model-based RL with zero correction and three corrections held fixed within each attempt: a constant offset estimated during warm-up, recursive least squares (RLS) updating a bias and gain after each attempt, and, on the real robot, fixed-gain feedback subtracting the preceding attempt's error. All share the correction channel, bounds, and 5 warm-up attempts; our actor adjusts the correction at every control step during the reach and hold, with its weights fixed within the attempt. Simulation also includes a model-free actor-critic acting at every control step, with the same correction increments, bounds, and exploration noise.

Model-based RL improves accuracy beyond offset estimation. Our learner keeps lowering the error after the offset estimators level off, ending at 1.3 cm against 2.1 cm for the constant offset, 2.5 cm for RLS, and 4.6 cm without a correction (Fig. 9a). From attempt 25 on, its median error is 1.5 cm against 2.2 cm and 2.7 cm, lower than both estimators in each of the three sessions ($p = 5 \times 10^{-4}$ against the constant offset, two-sided Mann–Whitney U test over the attempts of the three sessions). The model-free actor-critic, which also changes the command at every control step, ends at 3.5 cm under the same interaction budget, so acting at every step does not by itself bring the gain. We run three 50-attempt sessions per learner at a fixed simulated apple placement, interleaving zero-correction at-

tempts after warm-up; the error is the median $\|c_t - n_t\|$ over the pre-closure hold, and the constant offset, RLS, and our learner all reach about 2.6 cm right after warm-up.

The model learns more than a constant tracking bias. The learned model predicts held-out trajectories better than a constant bias: open loop from their first measured state, its error falls from 4.3 cm after five training attempts to 0.4 cm after 40, against 4.1 cm for the command plus a fitted constant offset.

Online correction improves pickup success. On seeded random placements in simulation, our learner raises first-try lift success from 47.8% without a correction to 72.2%, and success within ten attempts from 65.6% to 93.3%. With the simulator's exact targets and robot poses, the offset estimators reach similar rates.

The fast system achieves higher real-robot success. On real-robot floor pickups, our learner has the highest success on every object, 38/60 attempts in total versus 15/60 without a correction, and the lowest error on the can and apple (Tab. 3). Retrying in place then lifts all 36 eligible placements within five tries, up from 67% at the first try, while the control rises from 29% to 76% (Fig. 9b). We evaluate banana, can, and apple pickups with a three-finger end effector in separate sessions, scoring 20 attempts per method and object after learner warm-up. Success requires standing up holding the object; tracking error, the distance from the contact point to its target n_t (which the correction does not move), is measured at contact. Whether a grasp at that target holds also depends on perception and on the object, so error and success need not rank the learners alike. Contact and target calibrations vary across sessions, and our apple run also uses a camera correction, so these runs compare deployed configurations.

5 Conclusion

ATLAS is an adaptive harness between a pretrained vision-language agent and a frozen whole-body tracker. From one related human video and an instruction, it adapts the motion reference and learns from robot attempts to refine grounding and correct end-effector commands. Real-robot deployments demonstrate interactions from object pickup to stair traversal with manipulation, and reference reuse across objects and scenes. On SIMPLE, one human video yields 29/30, 28/30, and 30/30 successes, within one success of the best listed baseline on each task; the baselines use 100 robot demonstrations per task for fine-tuning. In fixed-placement simulation, online model-based RL ends at 1.3 cm median tracking error versus 3.5 cm for model-free RL under the same interaction budget. Across real-robot configurations with differing calibrations, our learner achieves the highest observed pickup success on all three objects. Agent review recovers simulated pickup from an incorrect object description; held-out replay of robot observations shows recall rising from 18.9% to 78.3% and fewer false success judgments. These behaviors use no task-specific robot demonstrations or offline task-policy training, and all pretrained models stay frozen during deployment.

Limitations. The learned dynamics model is specific to one task step and session, so it does not yet scale across tasks. Grounding from phrases and depth is also not yet precise enough for fine-grained interactions, which the pipeline cannot yet perform.

References

[1] Arthur Allshire et al. Visual imitation enables contextual humanoid control. *arXiv:2505.03729*, 2025.

[2] Marcin Andrychowicz et al. Hindsight experience replay. *arXiv:1707.01495*, 2017.

[3] Lars Ankile et al. Residual off-policy RL for finetuning behavior cloning policies. *arXiv:2509.19301*, 2025.

[4] Joao Pedro Araujo et al. Retargeting matters: General motion retargeting for humanoid motion tracking. *arXiv:2510.02252*, 2025.

[5] Zhihai Bi et al. GenHOI: Contact-aware humanoid-object interaction by imitating generated videos without task-specific training. *arXiv:2606.12995*, 2026.

[6] Douglas A. Bristow, Marina Tharayil, and Andrew G. Alleyne. A survey of iterative learning control. *IEEE Control Systems Magazine*, 26(3):96–114, 2006.

[7] Nicolas Carion et al. SAM 3: Segment anything with concepts. *arXiv:2511.16719*, 2025.

[8] Sirui Chen et al. CHIP: Adaptive compliance for humanoid control through hindsight perturbation. *arXiv:2512.14689*, 2025.

[9] Sitong Chen et al. KINO: A keyframe interface for VLM planning and whole-body control in humanoid loco-manipulation. *arXiv:2609.18869*, 2026.

[10] Cheng Chi et al. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems (RSS)*, 2023.

[11] Seoyeon Choi et al. HuGo: LLMs as whole-body policy code designers for humanoid loco-manipulation. *arXiv:2609.30594*, 2026.

[12] Zhaoyang Chu et al. OpenRUA: Robot-use agents are zero-shot visuomotor policies. *arXiv:2610.02459*, 2026.

[13] Marc Peter Deisenroth and Carl Edward Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *ICML*, 2011.

[14] Runpei Dong et al. HERO: Learning humanoid end-effector control for visual whole-body open-vocabulary object grasping. *arXiv:2602.16705*, 2026.

[15] Dyna Robotics. Dyna-2.1: A physical agent for end-to-end workflows. https://www.dyna.co/dyna-2.1, 2026. Blog post, 2026-09-29.
[16] Yuan Fang et al. Find something you can't do: Agentic real-world reinforcement learning for self-improving VLA models. *arXiv:2609.32069*, 2026.
[17] Figure AI. Introducing Helix 02: Full-body autonomy. https://www.figure.ai/news/helix-02, 2026. Blog post, 2026-01-27.
[18] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In *NeurIPS*, 2021.
[19] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *ICML*, 2018.
[20] Meng-Hao Guo et al. Transferring the intelligence of VLMs to robotic control. *arXiv:2609.22966*, 2026.
[21] Shuliang He et al. Praxis: Distilling physical interaction priors from egocentric videos for generalizable whole-body manipulation. *arXiv:2609.30735*, 2026.
[22] Shuliang He et al. RoboReact: Agentic skill distillation from generated egocentric videos for generalizable whole-body manipulation. *arXiv:2608.03387*, 2026.
[23] Wenlong Huang et al. ReKep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. In *CoRL*, 2024.
[24] Yuhao Huang et al. Online sim-to-real adaptation via closed-loop system modeling. *arXiv:2609.28878*, 2026.
[25] Gio Huh et al. HomeBody: A humanoid that explores, remembers, and acts on its own. https://tml.stanford.edu/homebody/, 2026. Project page, September 2026.
[26] Tobias Johannink et al. Residual reinforcement learning for robot control. In *ICRA*, 2019.
[27] Jinhan Li et al. OKAMI: Teaching humanoid robots manipulation skills through single video imitation. In *CoRL*, 2024.
[28] Yitang Li et al. BFM-Zero: A promptable behavioral foundation model for humanoid control using unsupervised reinforcement learning. *arXiv:2511.04131*, 2025.
[29] Jacky Liang et al. Code as Policies: Language model programs for embodied control. In *ICRA*, 2023.
[30] Qiayuan Liao et al. SuperMimic: Scaling humanoid whole-body control with one differentiable world model. In *CoRL*, 2026.
[31] Zeyu Ling et al. GenTrack: Physical alignment for robot-native motion generation and zero-shot humanoid tracking. *arXiv:2608.01410*, 2026.
[32] Zhengyi Luo et al. SONIC: Supersizing motion tracking for natural humanoid whole-body control. *Science Robotics*, 2026.
[33] NVIDIA. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv:2503.14734*, 2025.
[34] NVIDIA. GR00T N1.7 checkpoint for the Isaac Lab-Arena static pick-and-place task. https://huggingface.co/nvidia/GN1x-Tuned-Arena-G1-Static-PickNPlace, 2026.
[35] NVIDIA. Isaac Lab-Arena: simulation environments for policy evaluation in the GR00T end-to-end workflow. https://docs.nvidia.com/learning/physical-ai/gr00t-e2e-workflow/latest/, 2026.
[36] Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. In *NeurIPS*, 2018.
[37] Physical Intelligence. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv:2504.16054*, 2025.
[38] Zehong Shen et al. World-grounded human motion recovery via gravity-view coordinates. In *SIGGRAPH Asia*, 2024.
[39] Chen Tessler et al. Metatron: A promptable whole-body execution layer for humanoids. https://research.nvidia.com/labs/par/projects/metatron/, 2026.
[40] Caojun Wang et al. Anchor 1: Precise and stable humanoid control for whole-body manipulation. *arXiv:2611.00193*, 2026.
[41] Tingwu Wang et al. MotionBricks: Scalable real-time motions with modular latent generative model and smart primitives. *ACM Transactions on Graphics (SIGGRAPH)*, 2026.
[42] Yikai Wang et al. KPI: A promptable kernel for physical interaction on humanoids. *arXiv:2609.36151*, 2026.
[43] Songlin Wei et al. SIMPLE: Simulation-based policy learning and evaluation for humanoid loco-manipulation. *arXiv preprint arXiv:2606.08278*, 2026.
[44] Songlin Wei et al. Ψ_0: An open foundation model towards universal humanoid loco-manipulation. *arXiv:2603.12263*, 2026.
[45] Philipp Wu et al. DayDreamer: World models for physical robot learning. In *CoRL*, 2022.
[46] Weiji Xie et al. Test-time compute for humanoid behavior foundation models. *arXiv:2610.14896*, 2026.
[47] Seonghyeon Ye et al. World action models are zero-shot policies. *arXiv:2602.15922*, 2026.
[48] Jiarui Yuan et al. Democratizing robot-use agents: A unified harness with open models. *arXiv:2611.01560*, 2026.
[49] Weishuai Zeng et al. Scaling behavior foundation model for humanoid robots. *arXiv:2607.15163*, 2026.
[50] Siheng Zhao et al. ResMimic: From general motion tracking to humanoid whole-body loco-manipulation via residual learning. *arXiv:2510.05070*, 2025.
[51] Tony Z. Zhao et al. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems (RSS)*, 2023.

Appendix

This appendix first details the whiteboard writing of Sec. 4.3 (Appendix A) and then reports further experiments on each part of ATLAS. Appendix B ablates the fast system's learner one component at a time and tests it on a second task. Appendix C compares the harness with direct replay of the human motion and runs it on six whole-body trackers. It also compares harnesses written by five different agents. Appendix D evaluates the slow system on real-robot frames and online in simulation. Appendices E to G give the method details, and Appendices H and I the simulated and real-robot protocols.

A Writing on a whiteboard

Writing needs the marker on the board through every stroke, so the harness follows the board as the body moves. The marker leaves ink only while its tip touches the board, the word takes 1.1 m of strokes, and the standing body moves by up to 28 mm relative to the board within a single word. The harness measures the board's plane from the depth image of the robot's own head camera before every attempt and eight times per second while writing, and keeps the strokes' end-effector targets on that plane. The reference comes from the word's strokes: it holds a standing pose while the right wrist traces the 35 cm-wide word. The fast system acts within the board's plane and treats each letter as a separate attempt. From the first attempt on the board to the last, no one touches the robot: a person only starts each attempt and wipes the board. Figure 5 (last row) shows a later session, in which it writes "Atlas".

B Additional experiments: the fast system

Every experiment here runs the fast system (Sec. 3.4 and Appendix F) in Isaac Lab (Appendix H). The last two expand on the results of Sec. 4.6. An attempt's hold error is the median $\|c_t - n_t\|$ over its pre-closure hold, as in Fig. 9a.

To test whether each component of the learner is needed, we remove one at a time. Removing any one component of the learner raises its hold error (Fig. 10). We repeat the floor-pick sessions of Fig. 9a without each component, rerunning the full learner alongside. All sessions use the same placement and warm-up. Sessions at the same seed give each attempt the same mode: no correction, the actor, or the actor exploring (Appendix H).

- *No shoulder-relative model.* The model's inputs and output, and the actor's copies of these inputs, are taken relative to the floor point under the pelvis at the attempt's start instead of the moving shoulder b_t. The model must then also predict the body's motion.
- *Single model.* One member replaces the ensemble of 5 members.
- *No look-ahead commands.* The model's input keeps the current command and drops the look-ahead commands after it.
- *No goal augmentation.* The actor is trained on each replayed attempt with its own target only, without shifted targets.
- *No randomized prior.* The 5 members lose their fixed random prior and differ only in their initialization.
- *Actor without the errors ahead.* The actor's state keeps the current error $n_t - c_t$ and drops the errors 5 and 10 steps ahead.
- *Actor without the applied correction.* The actor's state drops the applied correction and keeps the correction in flight.
- *One-step model loss.* The model is fit with a one-step prediction loss from each measured state instead of the multi-step loss of Sec. 3.4.
- *No command history.* The model's input drops the 9 previous commands.
- *No phase.* The model's input drops the reference phase.

The actor still reads every input dropped from the model.

The full learner's hold error averages 1.44 cm over three sessions, and a single model is worse in all three, at 1.71 cm on average. Without the shoulder-relative frame, the model's held-out prediction error more than doubles, from 0.39 to 0.96 cm, and the mean hold error rises to 1.59 cm. Without the look-ahead commands, it rises to 1.83 cm, and without goal augmentation to 2.07 cm. Each of the six further variants, run once, also stays above the full learner's 1.44 cm. Without the reference phase, the error rises to 2.46 cm, and without the command history to 2.40 cm.

To test whether the learner carries over to another task, we run it on a standing apple pick. The learner cuts the hold error from 3.03 to 0.96 cm (Tab. 4). The task reuses the standing reference of Fig. 9c with the floor pick's apple and three-finger end effector. The apple rests on a 50 cm support. Every one of the twelve attempts in which the actor acts without exploration stays within 1.02 cm, including the first.

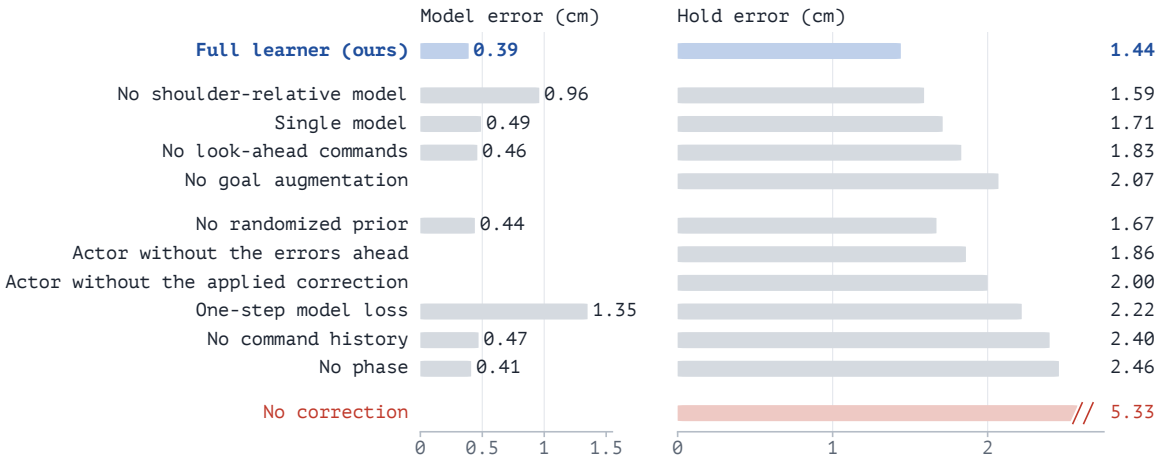


Figure 10. The full learner has the lowest hold error. Floor pick at one placement, in new sessions under the protocol of Fig. 9a. Hold error: median over a session's attempts in which the actor acts without exploration, averaged over three sessions; the last six variants ran one session. Model error: held-out prediction error after 40 training attempts (Tab. 5), for the variants that change the model. No correction: the attempts with no correction in the full learner's sessions.

	Hold error (cm) ↓	
Task	No correction	Ours
Floor pick (three sessions)	5.33	**1.44**
Standing apple pick (one session)	3.03	**0.96**

Table 4. The learner cuts the hold error on both tasks. Median hold error over the attempts with no correction and over those in which the actor acts without exploration, within the same sessions. Floor pick: the means of Fig. 10. On the standing apple pick, the robot localizes from the simulator's root pose, as in Fig. 9c.

	Error (cm) ↓ after k training attempts			
Prediction	$k = 5$	$k = 10$	$k = 20$	$k = 40$
Learned model (ours)	4.3	1.6	0.9	0.4
Command plus constant offset	4.3	4.2	4.1	4.1
Command alone	9.7	9.7	9.7	9.7

Table 5. The learned model predicts the end effector's path. Mean distance between the predicted and the measured end-effector position over a held-out attempt's reach and hold, predicted open loop from its first measured state. Each cell averages over a session's attempts after the 40th, then over the nine sessions. Constant offset: the mean of the measured minus the commanded position over the first k attempts.

To test whether the dynamics model learns more than a constant offset, we compare their predictions. After 40 training attempts, the learned model's open-loop prediction is off by 0.4 cm on average, a tenth of the error of a constant offset (Tab. 5). We refit the model offline with the learner's own code and schedule. The data are the apple sessions of Fig. 9a with our learner, the constant offset and recursive least squares, nine in all. For each session, the model trains on its first k attempts and is tested on its attempts after the 40th. From a test attempt's first measured state, it predicts the end effector's position over the whole reach and hold. It reads only the recorded commands and shoulder trajectory, with no measurement of the end effector in between. The error falls steadily with k. The command plus a constant offset fitted on the same attempts stays near 4.1 cm, and the command alone is off by 9.7 cm.

To test whether the dynamics model is needed, we compare with a model-free learner. The model-free actor-critic of Fig. 9a also acts at every control step, yet it ends at 3.5 cm against 1.3 cm for our learner (Sec. 4.6). It shares our learner's correction increment, rate, bounds and acted axes, its exploration noise, and the same mode for every attempt, drawn from the same seeds. In place of the dynamics model, it learns twin critics from the logged steps. It is an off-policy actor-critic in the style of TD3 [19], with multi-step returns, delayed actor updates, target-policy smoothing and a behavior-cloning term toward the executed increments [18]. It is updated after every attempt on all attempts so far, with targets shifted as in our goal augmentation. Its state is the phase, the error $n_t - c_t$, the applied correction and its change over the last ten control steps. Its reward is minus the error on the acted axes at every step of the reach and hold.

C Additional experiments: the harness and its agent

To test whether editing the motion is needed, we replay the human motion directly. Replayed directly, the retargeted human motion fails at the contact in all three tasks of Fig. 11, and ATLAS completes each of them. In this baseline, the frozen tracker plays the motion as recorded in the same Isaac Lab scene as ATLAS, with no perception and no edit of the reference. The replayed motion closes both end effectors beside the box, stops the gripper short of the cabinet's handle and closes the end effector beside the apple. ATLAS aligns the reach with the perceived contact targets and runs on ScaleBFM [49], our default System 0.

To test whether the harness depends on one tracker, we run it on six whole-body trackers. One fixed harness per task, with its specification and fast system, runs on each of them, and only System 0 changes (Tab. 6). Localizing onboard, ATLAS opens the cabinet door with every tracker and lifts the apple in a standing pick with five of them. It opens the door in all 60 scored attempts with Anchor 1 and with Metatron, and in 59 with ScaleBFM. On the door, the fast system lowers every tracker's hold error on the two axes it corrects, forward and sideways, from 2.4–7.5 cm before its correction to 0.2–2.8 cm after. Besides ScaleBFM, we test SONIC [32] and Anchor 1 [40], which tracks the pelvis, wrists and ankles of the reference (Fig. 12). We also test three recent releases: Metatron [39], CHIP [8] and HERO [14]. Metatron takes the torso, ankle and hand targets of the edited reference. CHIP takes the head and palm targets and the reference's leg joints. HERO takes palm goals through its own inverse kinematics, with its root position from leg odometry. For SONIC, we use joint targets on the door and the three-point input (head and both hands) on the apple. Each tracker runs three sessions on the cabinet door and one on a standing apple pick, which reuses the standing reference of Fig. 9c with the three-finger end effector; unlike the standing pick of Tab. 4, the apple rests on a 54.5 cm support and the robot localizes onboard. A door session has 5 warm-up attempts and 20 scored ones; an apple session has 45 attempts, of which the last 20 are scored. In the scored attempts, the actor acts without exploration.

To test whether the harness depends on one agent, we have five different agents write it. From one human video demonstration and one prompt, an agent writes a task's harness in about a minute of model calls, without revision (Tab. 7). It annotates the interaction frames and specifies the task, and the motion-editing settings stay fixed across agents. The 7 tasks span a floor pick, a box carry, a cabinet door, a candy taken from a basket, trash collected on the way to a sofa, and two stair tasks with a pick. Claude Opus 5.5 and Haiku 5.5 write an executable harness for all 7 tasks, the open-weights Qwen3-VL-32B for 6, Sonnet 5.5 for 5 and GPT-6.2 Astra for 5. In Isaac Lab with the fast system enabled, GPT-6.2 Astra's harnesses open the cabinet door in all 20 attempts after warm-up and lift the box in 60%. Opus's harnesses lift the box in 75% and open the door in 65%. Haiku's box harness lifts the box in 80% of attempts and Qwen's in 20%.

	Cabinet door		Standing apple	
System 0	Success ↑	Hold error (cm) ↓	Success ↑	Hold error (cm) ↓
Anchor 1 [40]	**60/60**	2.44 → **0.18**	**18/20**	1.89 → 0.76
Metatron [39]	**60/60**	2.41 → 0.60	13/20	3.46 → **0.59**
ScaleBFM [49]	59/60	2.78 → **0.18**	13/20	1.13 → 0.88
SONIC [32]	53/60	6.13 → 0.97	5/20	3.10 → 1.80
CHIP [8]	46/60	4.84 → 2.84	0/20	8.22 → 7.47
HERO [14]†	45/60	7.54 → 1.17	9/20	6.04 → 1.43

Table 6. The same harness runs on six whole-body trackers. Isaac Lab, with onboard localization; on each task, every tracker runs the same harness, with its specification and fast system. Success: the door opened at least 30°, or the apple lifted and held, over the scored attempts. Hold error, on the axes the correction acts on (forward and sideways at the door; forward, sideways and down at the apple): median over a session's warm-up attempts, before the fast system corrects, → median over its scored attempts; both averaged over sessions. †With its root position from leg odometry and the simulator's joint control in place of its own. Apple error: attempts with the apple on its support at contact.

To test whether the agents' output is stable, we regenerate the specifications deployed on our robot. Given only the prompt and the demonstrations, Opus reproduces the deployed specification for 25 of 28 held-out tasks, Sonnet for 24 and Haiku for 21. A task counts when the 3 independent generations agree and match the deployed specification in every key field (Tab. 12), with the segmenter phrases and the end state compared by meaning. The system prompt was fixed after development on four other tasks.

D Additional experiments: the slow system

The slow system fixes two failures that a command correction cannot: the segmenter misses the object because the phrase that names it does not fit the scene, or the robot judges an attempt a success although its end effector is empty. In both, the agent reads the robot's saved head-camera frames and rewrites text: the object phrase, or the check that judges success (Sec. 3.5). The first two experiments test each rewrite offline, replayed on later frames of real-robot runs, and the last

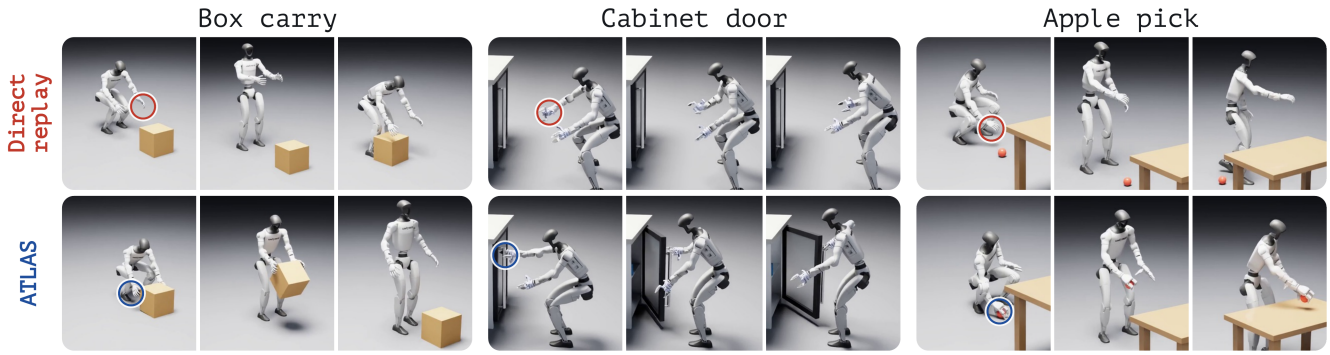


Figure 11. The harness completes the interactions that direct replay misses. Isaac Lab. The retargeted human motion replayed directly (top) and edited by ATLAS (bottom), at the demonstration's contact, midway and end. Rings mark the end effector missing (red) or making (blue) the contact.

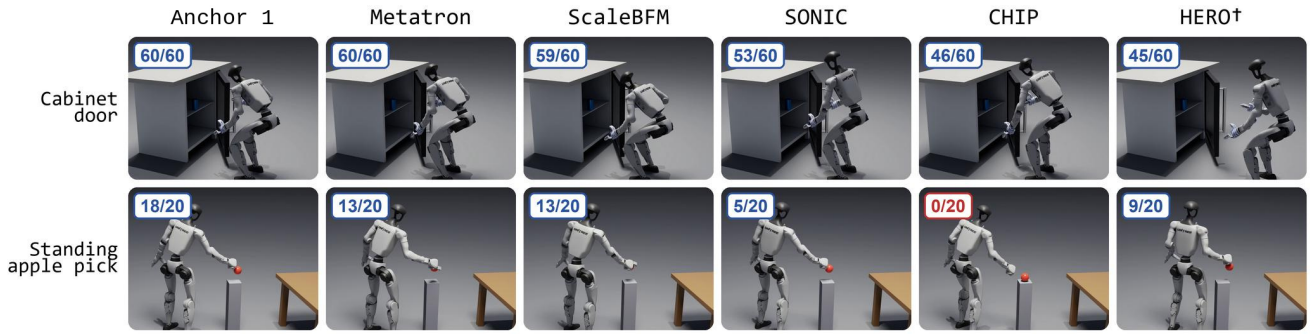


Figure 12. The same harness on each tracker. Isaac Lab, with onboard localization. Each cell shows one attempt at the end of the pull of the cabinet door (the door at 30°) or of the lift of the apple. The attempt is the first successful one after the 5 warm-up attempts of a separate session run to render these images, or the first after the warm-up if none succeeds. Badges: the tracker's success in Tab. 6, blue when the shown attempt succeeds and red when it fails. HERO† takes its root position from leg odometry.

two run the slow system online in simulation. The first three give the full results behind Sec. 4.4.

To test whether the rewritten phrase fixes missed detections, we replay it on later real-robot frames. With the phrase in force, the segmenter often missed the object in four real-robot runs. The agent proposes new phrases from each run's early frames and keeps the one that scores best on its own labels of those frames. The segmenter is then replayed with that phrase on each run's later frames, 180 in all with the object in view. Recall, the share of these frames on which the segmenter finds the object, rises from 18.9% to 78.3%, with wrong-object readings at 1.6% (1.1% with the phrase in force; Tab. 8a). A segmenter threshold tuned on the same early frames reaches 66.1%, below the agent's phrase (Holm $p = 0.049$), and a paraphrase written without the frames reaches 75.0% but boxes the wrong object on 5.4% of the frames. With Sonnet 5.5 or Haiku 5.5 as the agent, recall is 31.7% and 22.2% (Tab. 8b); Haiku keeps the phrase in force on three of the four runs. The segmenter finds the object when its box contains the object's point as labeled in Appendix I; a wrong-object reading is a box that misses that point, exceeds twice the object's extent, or falls on a frame without the object.

To test whether the rewritten success check fixes wrong judgments, we replay it on later real-robot attempts. The robot judges every attempt itself: a false success ends a trial that failed, and a false miss costs a retry. The agent rewrites the check from the earlier attempts of each real-robot run. Each check in Tab. 9 is then replayed on 103 later attempts and compared with blind labels, read afterwards from each attempt's frames and from the next approach's first look at the object's rest spot. The agent's check cuts false successes by two thirds, from 30 to 10 of the 72 attempts that held nothing, and disagreements with the labels from 39 to 24 (Holm $p = 0.025$; Tab. 9); false misses rise from 9 to 14 of the 31 attempts that held the object. A logistic regression fitted to the labels of the earlier attempts, and the agent without frames, suppress more false successes but reject most of the held objects.

Agent	Executable (of 7) ↑	Time (s) ↓	Box lift (%) ↑	Cabinet door (%) ↑
Claude Haiku 5.5	**7**	82	**80**	0
Claude Sonnet 5.5	5	25	0	no harness
Claude Opus 5.5	**7**	54	75	65
GPT-6.2 Astra	5	**14**	60	**100**
Qwen3-VL-32B	6	35	20	no harness

Table 7. Different agents write executable harnesses from one human video demonstration and one prompt. Each agent writes once, without revision; these single writes are separate from the fixed harnesses run in Tab. 6, and the motion-editing settings are fixed. Executable: passes the parsing, vocabulary and frame checks (within the recording and mutually consistent), with no refusal. Time: mean wall time of the agent's calls per harness. Box lift and cabinet door: success in Isaac Lab with our learner, over 20 attempts after 5 warm-up attempts; the box lifted and held, or the door opened at least 30°. No harness: the agent's door answer is not executable; Sonnet's cut of the video fails the frame check, and Qwen3-VL declines, judging the demonstration unfit for a door. Haiku's door harness pulls with the left hand, whereas the person pulls with the right, and opens the door in no attempt.

	Recall (%) ↑	Wrong object (%) ↓
(a) Who revises the phrase		
Phrase in force (control)	18.9	1.1
Threshold tuned	66.1	3.2
Paraphrase without frames	75.0	5.4
Agent with frames (ours)	**78.3**	**1.6**
(b) Which model is the agent		
Claude Haiku 5.5	22.2	**1.1**
Claude Sonnet 5.5	31.7	8.1
Claude Opus 5.5	**78.3**	1.6

Table 8. The agent quadruples recall with almost no wrong readings. Revisions come from early frames of four real-robot runs where the phrase failed and are scored on later ones (180 with the object in view). Bold best, underline second among the revising methods.

Success check	False success (%) ↓	False miss (%) ↓	Disagreements ↓
Rule in force	42 (30/72)	29 (9/31)	39
Logistic regression	7 (5/72)	68 (21/31)	26
Agent without frames	10 (7/72)	77 (24/31)	31
Agent with frames (ours)	14 (10/72)	45 (14/31)	**24**

Table 9. The agent's revised success check cuts false successes by two thirds. Checks replayed on 103 held-out real-robot attempts and scored against blind hindsight labels; the logistic regression is fitted to the labels of the earlier attempts. False success: a success judged among the 72 attempts that held nothing; false miss: a miss judged among the 31 that held the object; disagreements: their sum, of 103.

Session	Off, lifted	On, lifted	Cycles	Switch	After the switch, lifted
1	0/20	12/20	2	9	12/12
2	0/20	14/20	1	6	14/15
3	0/20	11/20	2	9	11/12
All	0/60	37/60	5	–	37/39

Table 10. The slow system rescues a pick that never succeeds without it. Lifted: attempts that lift the apple, by the simulator's state, of 20 per session and condition. Cycles: review cycles run with the slow system on; the last one keeps "the small red apple." Switch: the first attempt on the new phrase.

To test whether the slow system repairs a wrong object phrase online, we start a simulated pick with one. The phrase is "the small green apple.", while the scene's apple is red, so the segmenter does not find it (Fig. 13a). Without the slow system, nothing revises the phrase, and the robot lifts the apple in 0 of 60 attempts. With it, every session adopts "the small red apple." within two review cycles, and the robot lifts the apple in 37 of 60 attempts, 37 of the 39 after the switch; the retry in place recovers each of the two misses (Tab. 10). Each condition runs three sessions of 20 attempts with the fast system enabled, and the simulator's state scores every attempt. A review cycle takes 61 to 69 s while the robot keeps acting on the phrase in force, as on the real robot. The agent sees the prompt, the phrase in force, the segmenter's counts and saved head-camera frames, but never the simulator's state; a candidate phrase is adopted under the acceptance rule of Appendix G, at a Holm-corrected $p < 0.05$.

To test the slow system on a prompt that names none of the items, we have the robot clear a simulated floor of trash. With the slow system, the robot puts every item in the bin in every session, and nothing is picked up without it (Tab. 11). Under the prompt "Pick up the trash and put it in the bin.", the floor holds a red and white can, a yellow plastic mustard bottle and a bin. The human video demonstration is a banana pick and supplies only the motion. The prompt names no item, and the agent's first phrase, "the small crumpled piece of trash.", matches none of them (Fig. 13b). With the slow system on, the first review names the can, which goes into the bin at attempt 7, the first under the new phrase. The second review names the bottle, and the floor is clear after 13 or 14 attempts. With it off, the robot keeps that phrase, and all 30 attempts of every session end before the contact. On the real robot, the slow system replaces the phrase "the trash." with one phrase per object (Fig. 8).

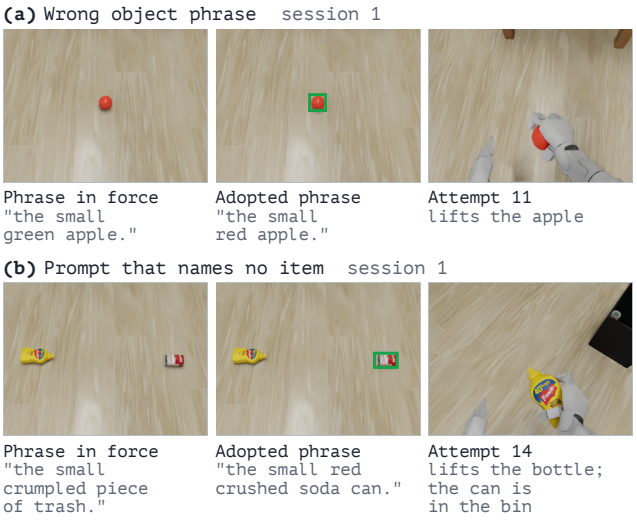


Figure 13. Under the phrase the review writes, the segmenter finds the object and the robot picks it up. Head-camera frames from session 1 of the two simulated runs. In each row, the first two pictures are one frame saved in attempt 2 and replayed by the review under the phrase in force and under the phrase it adopts; the box is the segmenter's. (a) The wrong object phrase (Tab. 10). (b) The prompt that names no item (Tab. 11).

Session	Off, cleared	On, cleared	Cleared by attempt	Cycles run/kept
1	0/2	2/2	7, 14	2/2
2	0/2	2/2	7, 13	2/2
3	0/2	2/2	7, 14	2/2
All	0/6	6/6	–	6/6

Table 11. The slow system clears a floor of trash. Cleared: items in the bin by the simulator's state, of 2 per session and condition. Cleared by attempt: with the slow system on, the attempts after which the first item is in the bin and the floor is clear. Cycles: review cycles run and kept with the slow system on. The kept phrases, in the order they take force: session 1, "the small red crushed soda can." and "the yellow plastic mustard bottle."; session 2, "the small red and white crushed can." and "the small yellow plastic bottle."; session 3, "the small red and white tin can." and "the small yellow plastic mustard bottle."

E Method details: the task specification and motion editing

From the video to the reference. The person's motion, recovered from the demonstration's video by GVHMR [38] and retargeted to the G1 by GMR [4], is cut by one model call. It reads video frames, one per 0.2 s, beside a table measured on the retargeted motion, one row per 0.04 s, and names the demonstrator's working hands, the grasp and release frames and the segments: a *pick* and, for a release in a later stance, a *place*, each by its *start*, *contact*, *settle* and *end*. The *held point*, the working end effector's contact point from grasp to release, is the path the affordance moves onto the object.

What the agent reads and answers. The agent answers once per task, before the first attempt, under a fixed system prompt and without tools. It sees the task's prompt, the robot's end effectors in words (fingers, a jaw or a rigid palm, and how each closes) and the demonstrations it may choose from as measured facts under neutral identifiers that name no object or task: the working hands, the action axes, the grasp and release stances, the held point's rise and the contact height. It answers in the closed vocabulary of Tab. 12 with a one- or two-sentence reason, or declines.

The agreement rule. A multi-demonstration request is first split into ordered steps, each a demonstration and a destination kind. The split and every step's setting are asked 3 times in isolated calls and compared by their *signature*: the split's ordered steps; a setting's demonstration, destination, contact relation and feature class, closure direction and axis, whether a force is given, and end state. The specification is written only when all 3 signatures agree, or a strict majority where the operator accepts one; otherwise a disagreement writes nothing. At test time only the slow system revises it (Sec. 3.5).

The segmenter phrases. SAM 3 [7] has not seen the scene, so each phrase gives the name and the look together, since either alone misses objects the other finds.

Placing the reference and aligning the wrist. At the handoff, the reference is turned in yaw and shifted on the floor so that its pelvis at the interaction start coincides with the robot's pelvis as the tracker reports it (Sec. 3.3). Only the interacting wrist's target receives the offset $p^\star - c_\tau^{\mathrm{ref}}$ of Eq. (1), from the demonstrated contact point to the perceived site; the tracker generates the whole-body motion from the edited reference. The wrist keeps the demonstrated orientation unless the object's geometry or the end effector calls for a turn.

Updates during the reach. Without external localization, the wrist target follows the object and the body during the reach (Sec. 3.3). Consistent head-camera readings re-anchor the site during the crouch, and the support leg's kinematics compensate the pelvis drift; both enter scaled by α_t. At the closure decision these updates stop and the shift reached so far is kept. The learned correction u_t is added on top, in the attempt frame (Sec. 3.4).

Field	Values, and what the robot does with each
demonstration	the identifier of one demonstration, or a decline
object, destination	a segmenter phrase each, *the* ⟨*color*⟩ ⟨*material or size*⟩ ⟨*common name*⟩., never where it sits; the object phrase names the part the end effector holds; the destination may be none, and with one the robot carries the object there and lets go at the demonstrated standoff
contact.feature	`face_centre`, `edge`: the end effector arrives on the upper face; `lateral_surface`: it closes around the object, the site is its measured center; `handle_inner_surface`, `bar`, `lip`: the center of the part the segmenter finds, two end effectors along a bar's length as far apart as the demonstrator's wrists
contact.relation	`opposite`: two end effectors on either side along the closing line; `same`: one, at its feature's site; `none`: every one at the measured center
closure.direction	`toward_each_other`: two opposite end effectors close until the object stops them; `away_from_each_other`: they move apart, nothing closing on the object; `along_axis_into_object`: a push or press; `along_axis_out_of_object`: a pull; `enclose`: fingers or jaw close around the object, the wrist still; `none`
closure.axis	one of the demonstration's action axes: `fore`, `grasp` (across the object) or `up`
closure.grip_force_n	newtons each end effector presses when the closure carries the object's weight, or null for a push, pull or press; it selects the progress indicator's geometric checks
outcome.end_state	one present-tense sentence of what the head camera sees once the motion has ended, if the request has succeeded
measured, not asked	the object's size, shape and axes and the closing line, from the mask and the depth when the robot looks

Table 12. The closed vocabulary of the task specification. Each value maps to one fixed behavior of the robot. The segmenter phrases, the site sentences (one per end effector) and the reason are free text and the grip force a free number; none of them enters the agreement signature beyond whether a force is given.

The progress indicator. With a grip force the indicator is geometric: the closure at the lift's end and whether the head camera sees the object above its rest height (Appendix G); without one, a closure on nothing ends the attempt at once. From the contact on, one VLM judge for every task, Qwen3-VL-4B, reads the head-camera frame with the task's words, and two consecutive looks naming a failure (the object lost or, in a carry or set-down, still where it rested) end the attempt. After the last frame it records whether the end-state sentence holds; success is scored as in Appendices H and I.

F Method details: the fast system

Window. In each attempt the fast system acts from reach onset (Sec. 3.3) through the hold at the contact to the step T that decides the close; the correction then ramps to zero (Tab. 13). The target n_t never contains the correction, so the learner cannot lower its error by moving its own target. The tracker's input describes each end effector by its wrist link, rigidly offset from the contact point where n_t and c_t are taken, and the correction shifts only that link's position in the input.

Plant and dynamics model. The plant maps the command, the harness's plus the applied correction, to c_t, a fixed contact point of the end effector, read by forward kinematics on the robot and from the simulated body's pose in simulation. The dynamics model splits it as $c_t = b_t + h_t$, takes the shoulder b_t as exogenous and replays it as recorded, and predicts h_t, the end effector relative to the shoulder. Each member of Eq. (3) sums a linear map, a SiLU network and a fixed random prior of the same structure [36], and predicts the change of h from the phase, the recent and look-ahead commands relative to b_t, and h_t with its last change. Each update fits the ensemble by a free run over every attempt's window from its first measured h, SuperMimic's multi-step loss [30], warm-started from the session's previous fit. On the real robot (Tab. 3), the fit also carries, per acted axis, a data-free prior $\mathcal{N}(1, 1)$ on the gain by which the end effector's hold position moves per unit of correction, so that the sign of the response is not decided by the few warm-up attempts alone; the session's own hold data update it.

Reward and actor. The reward of Eq. (4) takes the error only on the axes the correction acts on; its cost is the weighted squared change of the correction, in units of the rate. Per end effector, the actor π_ϕ of Eq. (2) reads the model's input, the error $n_t - c_t$ now and ahead, the applied correction and the correction still in flight, and outputs an increment squashed radially into the ball the tracker's correction channel admits per step; u_t sums the increments, clipped to the bound and to a length cap. It maximizes the undiscounted reward summed over the ensemble and over every past attempt that began at zero correction, reached the hold and never met the floor reflex, by which the correction channel holds back a downward correction once the end effector touches the floor. Each such attempt is replayed under the current actor, with its target shifted by a goal displacement from the session (Sec. 3.4) and its end-effector reading noised at the model's held-out one-step error. The gradient runs through the fixed model and through twins of the robot's squash, bound, cap, measured delay and correction channel,

one-sided where bound or cap saturate, so a saturated actor is pulled back without wind-up.

Warm-up, modes and updates. The first 5 attempts of a session are warm-up: a zero correction plus Gaussian exploration increments, the actor not read, so the dynamics model sees the correction act before the actor does. After them, in the sessions of Fig. 9a and of Fig. 10 and Tab. 4, a coin from the learner's own generator deals each attempt's mode: *zero* (no correction, the control) with probability 0.33, *mean* (the actor without exploration) with 0.33, and *explore* (the actor exploring) otherwise. In Tab. 3 and at the random placements (Appendix H), no coin is dealt: every attempt after the warm-up runs the actor without exploration, and the control runs in sessions of its own. Every attempt but a fall, first or retry and whatever its mode, trains the learner. The robot sends its logged steps to the PC, which updates both networks in the background on all the attempts so far; the robot never waits for it. An update is accepted if its parameters are finite and the twin of the correction channel left every step of the model's rollouts unchanged. Every attempt starts with the newest accepted actor, under the same hard bound, rate and cap as in the model's rollouts. No success label or held-out score selects an update. The correction acts forward and sideways and, on the floor picks of Tab. 3 and Fig. 9a, at the random placements (Appendix H) and in the sessions of Fig. 10 and Tab. 4, downward as well (Tab. 13).

G Method details: the slow system

What the agent reads. The slow system is the VLM agent of Sec. 3.2, running on the PC beside the robot (Sec. 3.5). For each attempt it receives one row the robot measures itself: the progress indicator's outcome, where the walk ended, the object's measured extent, the end effector's gap to the object at the close, the contact miss $c_T - n_T$, and the correction with its bound hits. It also sees success tabulated by first attempt and retry, by whether the correction acted, and by contact miss, gap and perceived height; the 12 newest successes and failures each; and the close and lift frames of the 2 newest of each, never the simulator's state.

When it acts. The agent proposes at most one revision at a time, never while a scored one is pending; the robot keeps acting on the specification in force while the agent works and loads the newest phrase before each attempt, so an accepted phrase takes effect at the next attempt (Sec. 3.5).

What it may change: the segmenter phrases. The object phrase, the phrase of the part the end effector holds (a handle bar, for example) and the destination phrase stay in the specification's template, "the <color> <material or size> <common name>.", with no location words. The agent is shown the frames where the object was in view but unfound or its mask fell elsewhere, writes two or three alternatives, each with the reason it reads from the frames, and the code scores them. *Recall* is the share of frames with the object in view whose mask's bounding box contains the object's point, taken from the end effector's contact point by forward kinematics at a success or from a point a VLM labeler marks on the frame, with an extent at most twice the object's. *Wrong object* counts masks on the robot's own links, projected into the head camera by forward kinematics, on the operator, or with an extent above twice the object's. The incumbent and each candidate are replayed on the same saved frames, with fresh segmenter sessions; in the simulated closed loop (Tab. 10), a candidate is kept only if its recall on the frames it was not proposed from rises, by an exact McNemar test on the discordant frames, Holm-corrected over the candidates, and its wrong-object rate does not. On the real robot (Fig. 8), the agent writes one phrase for each object of the task it sees, each checked on the saved frames, and the harness grounds them in turn; a revised phrase is kept if its detections of the object, net of wrong-object detections, exceed those of the phrase in force.

What it may change: the progress indicator. The indicator's rule says which evidence counts, closure at the lift's end, head-camera readings above the rest height, a reading inside the end effector, the palm gap in a carry, and how they combine; its VLM query is the end-state question, asked only where the head camera can see the answer. Each attempt gets a hindsight label from its close, lift and carry frames and the next approach's first look at the rest spot: a *false success* is an empty end effector in the carry or the object still at its rest spot at the next look, a *false miss* the object in the end effector after a judged miss. A revision is replayed deterministically on every logged attempt, the judge re-asked on the saved frames, and kept only if, on the attempts it was not proposed from, false successes do not rise and disagreements fall; a false success loses a whole trial, a false miss costs one retry.

H Simulation protocol

Isaac Lab in place of the robot. Simulated runs execute the robot's harness and learner code on the simulator's own runner, with nothing pretrained for the task. The simulator replaces the body and its sensors: a G1 with the task's end effector (for the Dex3 fingers, the contact point is the centroid of the closed finger-

Part	Setting	Value
Window	hold at the contact; ramp to zero after the close	≥0.5 s; ≥10 control steps
Dynamics model	members; hidden layers (SiLU); inputs per end effector	5; 2 × 64; 52
	look-ahead frames read; previous commands read	6; 9
	Adam steps per update; learning rate	300; 10^{-3}
Reward	error unit; weight of the correction's change	2 cm; 0.1
Actor	hidden layers; inputs per end effector	2 × 64; 62 (67 with the downward axis)
	error read ahead	5 and 10 control steps
	Adam steps per update; learning rate	200; 3×10^{-4}
Limits	bound forward; sideways; downward; rate	12 cm; ±15 cm; 5 cm; 5 mm per step
Exploration	increment s.d. per step and axis	2 mm (0.4 of the rate)
Modes	warm-up attempts; zero / mean probability	5; 0.33 / 0.33

Table 13. The fast system's settings, on the robot and in simulation. In the sessions dealt by a coin, the rest of its probability goes to exploration.

tips), the rendered head camera and depth image, the joint state and the root pose. A pick succeeds if the object ends the lift at least 5 cm above where it lay and within 12 cm of the end effector's contact point. This ground-truth score, not the robot's progress indicator, decides the retry.

Floor picks. The simulated floor picks take an apple (Fig. 9a and Fig. 10) or, at the random placements (below), a banana, a can or an apple off flat ground with the Dex3 fingers on the G1's right wrist. All three use one floor-pick demonstration, in which a person walks up, squats, takes a banana off the floor with the right hand, carries it and sets it down; the can and the apple each get their own segmenter phrase and affordance.

Cabinet door, box carry and standing apple pick (Tabs. 6 and 7). The cabinet, the box and the apple's 54.5 cm support stand at fixed places in their scenes, and the robot walks from its start to its station. It finds the handle, the box or the apple with the head camera and the segmenter. At the door and the apple it localizes by its own odometry, and at the box from the simulator's root pose. In Tab. 6, the robot reads no pose from the simulator. Every door and apple attempt starts from the reset scene, and a missed box lift is retried in place. The door succeeds when its hinge opens at least 30° within the demonstrated pull, and the box when it is lifted and held between the palms; the apple follows the pick rule above. On the apple, the correction also acts downward, by up to 10 cm. While the open hand holds at the contact, the simulator keeps the apple where it is, and when the fingers close it attaches the apple to the end effector if the contact point is within one apple diameter of the apple's center.

The learners. All act through the fast system's window, bounds and rate (Sec. 3.4). Let e_i be the mean end-effector error $c_t - n_t$ over the hold of attempt i and u_i the correction applied in it. The *constant offset* sends $u = -\overline{(e-u)}$ for the rest of the session, the bar being the mean over the warm-up attempts. *Recursive least squares* fits $e_i = \beta + G\,u_i$ per axis, a bias and a gain, after every attempt and sends $u = -\hat{\beta}/\max(\hat{G}, 0.3)$ on the next, clipped to the correction's bounds. Its priors, noise and forgetting $(1 - 1/80)$ are set from the robot's measured execution statistics. *Fixed-gain feedback*, run on the robot only, sends $u_{i+1} = u_i - e_i$, the last correction minus the last error, as if the gain were one. All share the fast system's warm-up and hold their correction constant within an attempt. The *model-based* learner is the fast system, unchanged. The *model-free actor-critic*, in Fig. 9a only, shares the fast system's correction increment and per-attempt modes and learns twin critics in place of the dynamics model (Appendix B).

One placement, perceived (Fig. 9a). Here the apple lies 0.6 m ahead of the robot, at the same place for every attempt, and the scene is reset before each one. The robot finds the apple with the segmenter and localizes by its own odometry, as on the real robot. A session is 5 warm-up attempts and 45 more. A coin assigns each of these our learner's correction, no correction or the correction with exploration, with probability 1/3 each; for the estimators it assigns their correction or none, with probability 1/2 each. Each learner runs three sessions. Figure 9a plots the median end-effector error over the hold, pooled over the sessions in bins of five attempts; the control curve pools the attempts assigned no correction in the nine sessions of our learner and the estimators. The sessions of Fig. 10 follow the same protocol.

Random placements (Sec. 4.6). Each placement puts the object on the floor 0.30 to 0.50 m ahead of the pelvis and 0.02 to 0.18 m to its left, turned by up to 30° either way. Placements are drawn from the session's seed, so every learner meets the same ones. The grasp target and the robot's pose are read from the simulator's exact state, and the robot does not walk. A session is

5 warm-up placements of one attempt each, then ten scored placements, and the learner carries over from one placement to the next. A miss is retried in place, and the scene is restored to the placement after five misses in a row or when the object leaves the placement area. A placement ends at the first lift or after ten attempts. Each learner and the control run three sessions per object. Section 4.6 reports the shares of scored placements lifted at the first attempt and within ten attempts, with the three objects weighted equally.

Object height (Fig. 9c). The object stands on a static block whose top is the given height above the floor, 12 cm square under the apple and 25 cm square under the can; at 0 cm the apple lies on the floor. The floor pick takes the apple with the three-finger end effector at 0, 5, 10, 20, 30, 40, 50, 60, 70 and 80 cm, and the standing pick takes a 330 ml can with the jaw gripper at 20, 30, 40, 45, 50, 60, 70 and 80 cm. Each height is one session of 15 attempts with a fresh learner: the first 5 are the fast system's warm-up and the other ten are scored. The robot finds the object with the head camera and the segmenter and localizes from the simulator's root pose; the fast system acts forward and sideways. The harness adapts the height as in Sec. 3.3, with the knees also kept at their demonstrated heights.

I Real-robot protocol and hardware

Robot and sensing. The real-robot experiments (Tab. 3) run on a 29-DoF Unitree G1 with a Dex3 three-finger end effector on the right wrist. All sensing is onboard: the built-in Intel RealSense RGB-D head camera and the joint state, which gives c_t by forward kinematics.

What runs where. The robot's Jetson Orin NX runs the 50 Hz control loop, in which the whole-body motion tracker takes the harness's command plus the correction as its end-effector command, and beside it the actor, which computes the correction of Eq. (2) at every control step, together with the end-effector driver and the retry. A PC on the local network runs the VLM agent, the segmenter [7], which reads the head camera at a fraction of the control rate, and the learner's updates, which run in the background and hand the newest actor to the robot.

Attempts, episodes and sessions. An attempt is one pass through the task's steps (Sec. 3); it ends at the end state, at a fall, at a 3-minute budget on any phase, or at a miss the robot detects itself (Sec. 3.1) and retries in place. An episode is the attempts on one placement of the object, up to its end state, cut short by a fall, by an object never seen from the robot's stance or by the cap of ten attempts. A session is one launch of the robot's program; each task step has its own learner, which starts from zero at the step's first entry and is kept across the session's episodes.

Learner and control. Each learner and the control run sessions of their own, and a learner's 5 warm-up attempts are not scored. An attempt succeeds when the robot stands up holding the object, by its own lift check, confirmed by the operator.

Placements (Fig. 9b). A placement runs from the operator placing the object to a lift; the eligible placements of Fig. 9b lie within the scored attempts of Tab. 3, except any begun before them or cut off before their fifth try.

Labels and endpoints. For the slow system's offline evaluation (Tabs. 8 and 9), Claude Opus 5.5 labels every saved frame from the task's words alone: whether the object is in view and where, and for an attempt's frames whether the end effector holds it. It never sees a phrase or a condition. Its labels of what the end effector holds are right on 92 of 105 simulated attempts, judged against the simulator's state. A second labeler, Qwen3-VL-4B, agrees with it on visibility on 582 of 607 frames ($\kappa = 0.70$) and on 181 of the 186 held-out frames of Tab. 8 ($\kappa = 0.63$). Every phrase comparison keeps its direction on the frames both labelers agree on. Opus thus selects by the labels that score it, while Sonnet and Haiku select by their own. The end-effector error, $\|c_t - n_t\|$ at contact, takes c_t from the joint state and the target n_t, both in the attempt's frame, and needs no label.

What a person does. No one teleoperates the robot. Within an episode no one touches or resets it, and a person only sets the object back in view when it leaves the camera's field of view; after a success, a person places the object for the next episode.