LEARNING TO ACT UNDER VISUAL INTERRUPTIONS WITH VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action (VLA) models have demonstrated strong capabilities in robotic manipulation, but they are typically developed and evaluated with all camera streams available throughout task execution. When a camera stops delivering frames during task execution, the policy must continue acting without access to subsequent observations from the missing view. Despite its practical importance, how such interruptions affect closed-loop manipulation remains insufficiently understood. To investigate this problem, we introduce **MAIL-Bench**, a benchmark that evaluates visual interruptions with VLA models. By interrupting different cameras at multiple stages of each policy's successful reference trajectory, MAIL-Bench measures how well policies retain their capabilities when visual inputs become unavailable. Building on this benchmark, we propose **MINT**, which trains VLA policies to handle missing visual inputs and temporarily supplies predicted images when cameras fail. It uses optical-flow extrapolation for missing wrist views or all-vision loss, and an action-conditioned world model for missing third-person views, withdrawing the predictions when they become unreliable. Experiments on $\pi_{0.5}$ and GR00T N1.5 show that MINT improves task success under camera loss over the original models. Experiments on the AgiBot G2 further demonstrate real-robot closed-loop deployment under camera loss.

1 INTRODUCTION

Vision-language-action (VLA) models (Brohan et al., 2023; Octo Model Team et al., 2024; Kim et al., 2025; Black et al., 2025b;a; NVIDIA et al., 2025; Xu et al., 2026) use visual observations and language instructions to perform robotic manipulation. Standard evaluations (Liu et al., 2023; Mees et al., 2022; James et al., 2020; Nasiriany et al., 2026) typically assume that all camera streams remain available throughout task execution. In practice, however, one or more camera streams may become unavailable while a robot is performing a task. Different cameras provide different views of the workspace: wrist cameras capture details near the robot's gripper, while third-person cameras provide a broader view of the scene. When a camera becomes unavailable, the remaining views may not provide the information previously available from that camera. Unlike changes in lighting or camera pose (Pumacay et al., 2024; Fei et al., 2026; Wang et al., 2025), these visual interruptions remove a source of information altogether. This motivates a research question: *How well can VLA models perform manipulation when some or all of their camera inputs are unavailable?*

Existing robustness benchmarks (Pumacay et al., 2024; Chen et al., 2026; Fei et al., 2026; Wang et al., 2025; Hendrycks & Dietterich, 2019) primarily examine changes in visual appearance, camera pose, or image quality while keeping camera streams available. Missing camera inputs pose a different problem: a policy no longer receives observations from one or more of its original camera views. The impact may vary across cameras and tasks, depending on what information remains available from the other views. Yet existing robustness evaluations provide limited insight into how VLA models perform under these conditions.

To investigate this problem, we introduce **MAIL-Bench**, a paired, closed-loop benchmark for evaluating VLA models with missing camera inputs. Built on RoboCasa365 (Nasiriany et al., 2024; 2026), it covers 18 manipulation tasks and 900 frozen scenes. Rather than removing cameras individually, MAIL-Bench evaluates the loss of distinct visual roles: the wrist view, both third-person views, and all vision. For each policy, we first identify the scenes it can complete with all cameras

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

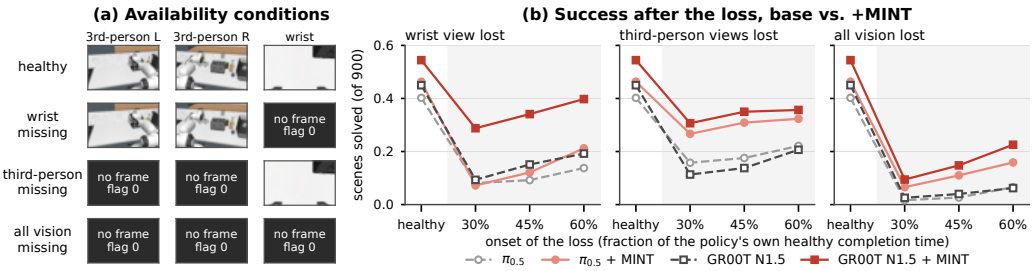

Figure 1: **MAIL-Bench.** (a) Four availability conditions on RoboCasa365; an interrupted camera delivers no frame and its availability flag is 0. (b) Scenes solved out of 900 with every camera present (healthy) and after one visual role is lost at 30, 45 or 60% of each policy's own completion time: camera loss removes a large share of both base policies' successes, and MINT recovers part of that loss on every role.

available and use its successful rollouts as references. We then remove the relevant camera inputs at 30%–60% of the policy's own completion time and keep them unavailable for the remainder of the episode. Missing frames are delivered as absent rather than replaced by the benchmark with blank or frozen images. By comparing paired rollouts in the same scenes and measuring success conditional on intact-camera success, MAIL-Bench quantifies how much capability a policy retains when visual inputs become unavailable.

Training VLA models with missing camera inputs (Skand et al., 2025; Vanjani et al., 2025) enables them to continue acting without the unavailable views, but it cannot recover the visual information those views would have provided. Predicting missing views with action-conditioned video models (Guo et al., 2026; Bardhan et al., 2026) offers a way to temporarily supplement the remaining observations, although prediction errors may accumulate as the robot and scene change. We therefore propose **MINT**, which combines training for missing camera inputs with selective visual prediction. During fine-tuning, MINT explicitly marks unavailable cameras and excludes their visual tokens, allowing the policy to act using the inputs that remain. At inference time, it uses optical-flow extrapolation for missing wrist views or all-vision loss, and an action-conditioned world model (Bardhan et al., 2026) for missing third-person views. Predictions are retained only while checks based on available observations and commanded actions are satisfied. Otherwise, they are withdrawn, and the policy continues with the corresponding inputs marked as missing.

Experiments on $\pi_{0.5}$ (Black et al., 2025a) and GR00T N1.5 (NVIDIA et al., 2025) reveal that the impact of missing camera inputs depends on which views are unavailable: losing a single wrist camera can be more disruptive than losing both third-person cameras. MINT improves task success under missing camera inputs on both models and, on GR00T N1.5, outperforms a matched clean fine-tuning control with the same training budget. We further deploy MINT on an AgiBot G2, where the same stack runs in real time and the robot completes manipulation tasks in closed loop with a camera interrupted partway through the task.

2 Related Work

Robustness evaluation of manipulation policies. Generalist suites measure competence with intact observations (Liu et al., 2023; Nasiriany et al., 2024; 2026; Mees et al., 2022; James et al., 2020; Li et al., 2023; Tao et al., 2025), and perturbation benchmarks (Pumacay et al., 2024; Chen et al., 2026; Fei et al., 2026; Wang et al., 2025; Li et al., 2025) change appearance, geometry, language, camera pose or the simulation-to-real gap, with every frame still delivered. Common-corruption suites (Hendrycks & Dietterich, 2019) degrade image content in the same sense, and a comparative study weighs world action models against VLAs under such perturbations (Zhang et al., 2026). MAIL-Bench instead evaluates the complete loss of a visual stream during an ongoing rollout, rather than a change to the content of an available frame. A camera stops delivering frames altogether, at a moment defined by the evaluated policy's own healthy trajectory rather than at a fixed step. The loss is applied per visual role rather than per device, the healthy and interrupted rollouts of a scene are paired by construction, and the outcome is scored conditional on healthy success (Table 1).

Table 1: Comparison of visual-robustness evaluation settings. MAIL-Bench uniquely targets persistent visual-stream loss during closed-loop execution.

Benchmark	Visual failure	Onset	Post-loss control
COLOSSEUM (Pumacay et al., 2024)	Perturbation	–	–
RoboTwin 2.0 (Chen et al., 2026)	Perturbation	–	–
VLATest (Wang et al., 2025)	Perturbation	–	–
LIBERO-Plus (Fei et al., 2026)	Black frame	Start	Closed loop
Missing-modality IL (Ismkhan & Bouchahcia, 2026)	Camera dropout	Start	Open loop
MAIL-Bench (ours)	**Camera loss**	**Mid-episode**	**Closed loop**

Learning with missing modalities. Prior work improves robustness to absent sensors through masked multimodal training, camera dropout, missing-state inference, retrieval and availability-aware representations (Skand et al., 2025; Vanjani et al., 2025; Xu et al., 2025a; Vogt-Lowell et al., 2026; Ismkhan & Bouchahcia, 2026; Wang et al., 2023; Kim & Hwang, 2026; Xu et al., 2025b); MINT's fine-tuning stage follows the masked-training line without adding a component to the policy. HAMLET (Koo et al., 2026) augments a pretrained VLA with learned history, whereas MINT intervenes upstream by determining what occupies a missing visual slot. Run-time observation interventions and failure-recovery methods (Hancock et al., 2025; Song et al., 2026; Lin et al., 2026; Ma et al., 2026; Yan et al., 2026) motivate selective inference-time correction; MINT's router is such an intervention, with the generated view supplied by optical flow (Farnebäck, 2003) or by an action-conditioned world model (Guo et al., 2026; Bardhan et al., 2026).

3 MAIL-BENCH: MANIPULATION UNDER VISUAL INTERRUPTION

We introduce MAIL-Bench, a paired, closed-loop benchmark for evaluating how visual-stream interruptions affect manipulation policies on tasks they can otherwise complete.

3.1 BENCHMARK CONSTRUCTION

Environment. We build MAIL-Bench on RoboCasa365 (Nasiriany et al., 2026), which provides diverse kitchen environments and everyday manipulation tasks. We select its 18 Atomic-Seen tasks to reduce the confounding effect of task generalization and focus on camera-loss robustness in previously learned tasks. With 50 fixed scenes per task, MAIL-Bench comprises 900 evaluation scenes.

Availability conditions. To make camera-loss conditions comparable across policies with different camera configurations, we group cameras by their visual roles rather than by individual devices. The wrist camera forms the *wrist* group: it moves with the gripper and observes the manipulated object at close range. The onboard cameras form the *third-person* group: they are fixed to the robot base and provide broader views of the arm and workspace. Then we consider four availability conditions. *Healthy* keeps all cameras available and serves as the reference rollout. *Wrist missing* removes the wrist view, *third-person missing* removes all onboard views, and *all vision missing* removes all visual input, leaving only proprioception and the instruction. We use *hard missing*: once interrupted, a camera group provides no further frames and its availability flag is set to 0.

Interruption timing. Camera loss may occur at different points during task execution. To evaluate this variation, we introduce interruptions at 30%, 45%, and 60% of each policy's successful healthy completion time. Defining onset relative to the policy's own execution accounts for differences in execution speed, rather than imposing the same absolute control step on every policy. Together, the three camera-loss conditions and three onset times define nine interruption settings, evaluated against a healthy reference.

3.2 BENCHMARK EVALUATION

Evaluation protocol. For each scene, we save the episode metadata and a settled simulator state immediately before policy execution. This state serves as the common starting point for the healthy rollout and all interruption rollouts. For each policy configuration, we first execute a healthy rollout with all cameras available. If the rollout succeeds, we record its completion step, T_{healthy}; otherwise, the scene is excluded from interruption evaluation.



Figure 2: A successful healthy rollout defines the reference completion time T_{healthy}. The same scene is then replayed with the selected visual role interrupted at 30%, 45%, or 60% of T_{healthy} and kept unavailable thereafter, while unaffected cameras continue streaming. Shown is an AgiBot G2 drawer example.

Each interruption rollout restarts from the same saved initial state as the healthy rollout and uses the same policy random seed. It receives normal camera observations until the designated interruption control step, which, for onset fraction $f \in \{0.30, 0.45, 0.60\}$, is defined as

$$T_f = \min(T_{\max} - 1,\ \lfloor f \times T_{\text{healthy}} + 0.5 \rfloor)\,, \tag{1}$$

where $T_{\max}$ denotes the task horizon.

Let $\mathcal{R}$ denote the set of visual roles. The policy receives observations only when it is queried, every few control steps; for each role $r \in \mathcal{R}$, let I_q^r denote its camera observation at the q-th query and $m_q^r \in \{0, 1\}$ indicate whether it is available. When a camera becomes unavailable at T_f, the benchmark discards the remaining actions of the current chunk and queries the policy immediately; we denote this query by q_f. Before q_f, $m_q^r = 1$; for an interrupted role, from q_f onward $m_q^r = 0$ and the visual input is replaced by $\varnothing$, indicating that no fresh frame is provided.

After the interruption, subsequent frames from the interrupted camera group are withheld from both the policy and any recovery module. Any replacement view must therefore be generated causally from information available at runtime, such as pre-interruption observations, surviving camera streams, and executed actions, without access to future simulator frames or states. Each successful healthy rollout gives rise to nine fault rollouts, covering three camera-loss conditions and three interruption onsets. Deterministic execution settings and checks for pre-interruption divergence are used to reduce variation unrelated to the camera intervention.

Evaluation metrics. We measure robustness only on scenes that the policy can already solve with all cameras available. For task j, let $n_j = 50$ be the total number of scenes, N_j^H the number solved under the healthy condition, and N_j^c the number of those same scenes that remain successful under interruption condition c. We define

$$H_j = \frac{N_j^H}{n_j}, \qquad M_{c,j} = \frac{N_j^c}{N_j^H}, \qquad S_j = \frac{1}{10}\Big(H_j + \sum_{c=1}^{9} M_{c,j}\Big), \qquad S_{\text{MAIL}} = \frac{1}{18}\sum_{j=1}^{18} S_j. \tag{2}$$

Here, H_j measures healthy task success, while $M_{c,j}$ measures how much of that capability is retained after interruption c. Conditioning $M_{c,j}$ on healthy success prevents robustness from being dominated by a policy's baseline task competence. For each task, healthy performance and the nine interruption conditions are weighted equally, and the final S_{MAIL} score averages equally over all 18 tasks. We evaluate every interruption condition even if a policy does not normally rely on the corresponding camera, since remaining unaffected by its removal is itself evidence of robustness. If a policy has no healthy success on a task, that task receives a score of zero.

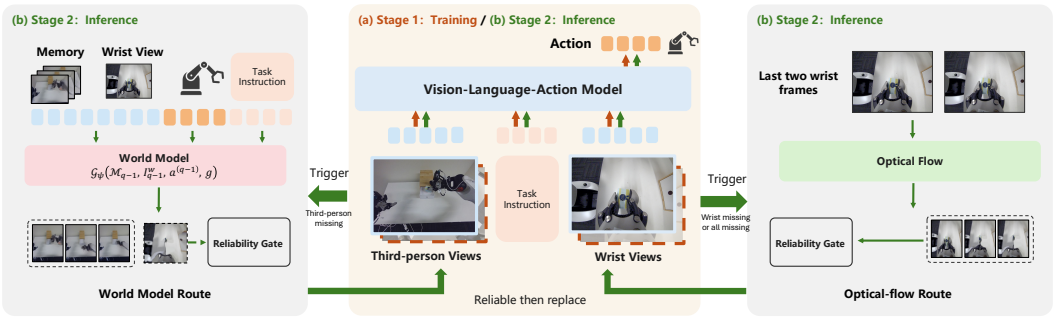

Figure 3: **MINT overview.** MINT separates robustness to missing visual inputs from visual prediction. The policy is first trained to act under variable camera availability, then selectively receives predicted views at inference time when a camera is interrupted. Unreliable predictions are withdrawn, and the policy falls back to acting from the observations that remain.

4 MINT: Acting under Visual Interruption

A robust VLA policy should remain functional when a camera becomes unavailable, without relying on the missing view being reconstructed. At the same time, when that view can be predicted reliably, the prediction may temporarily recover useful information that is no longer directly observed. MINT separates these two capabilities: learning with missing visual inputs teaches the policy to act from the observations that remain, while selective visual prediction supplements a missing view only when its prediction is considered reliable.

At the q-th query, given visual observations $\{I_q^r\}_{r\in\mathcal{R}}$, proprioceptive state s_q, and task instruction g, the policy predicts an action chunk of h actions

$$a^{(q)} \sim \pi_\theta\big(\cdot \mid \{I_q^r\}_{r\in\mathcal{R}},\, s_q,\, g\big)\,, \tag{3}$$

where h denotes the action horizon; only the first few actions are executed before the next query. Under visual interruption, one or more roles become unavailable during execution. MINT first adapts the policy to such changes in observation availability and then, at inference time, selectively supplies predicted observations for missing roles when reliable estimates are available.

4.1 Learning with Missing Visual Inputs

We first train the policy to predict actions when only a subset of its usual visual observations is available. When When $m^r = 0$, the corresponding view is marked as missing and its visual tokens are excluded from attention, matching the hard-missing semantics of MAIL-Bench.

Each training sample is a demonstration frame with observations $\{I^r\}_{r\in\mathcal{R}}$, state s, instruction g, and the next h demonstrated actions a^*. An availability pattern $m \sim \mathcal{C}$, sampled over the demonstration episode, determines which roles are missing at that frame, and $\tilde{I}^r$ denotes the input of role r under m. Using the backbone's original action objective ℓ, we optimize

$$\mathcal{L}(\theta) = \mathbb{E}_{(\{I^r\},s,g,a^*)\sim\mathcal{D},\, m\sim\mathcal{C}}\left[\ell\Big(\pi_\theta\Big(\cdot \mid \{(\tilde{I}^r, m^r)\}_{r\in\mathcal{R}},\, s,\, g\Big)\,, a^*\Big)\right], \tag{4}$$

where $\mathcal{C}$ is a distribution over temporal camera-availability patterns. It includes healthy, wrist-missing, third-person-missing, and all-vision-missing conditions, with both persistent and intermittent interruptions.

The policy architecture, optimization schedule, and action objective are otherwise unchanged. The resulting policy can therefore continue acting from the observations that remain when no reliable estimate of a missing view is available.

4.2 Predicting Missing Visual Inputs

Learning with missing visual inputs enables the policy to continue acting when a camera role is unavailable, but it cannot recover information that is no longer observed. At inference time, MINT

therefore constructs a candidate prediction for a missing role and supplies it to the policy only while the prediction remains reliable.

Let $\hat{I}_q^r$ denote a candidate prediction for visual role r at query q, and let $c_q^r \in \{0, 1\}$ indicate whether that prediction is currently accepted. The observation presented to the policy is

$$(x_q^r, \bar{m}_q^r) = \begin{cases} (I_q^r, 1), & m_q^r = 1, \\ (\hat{I}_q^r, 1), & m_q^r = 0 \ \wedge \ c_q^r = 1, \\ (\varnothing, 0), & m_q^r = 0 \ \wedge \ c_q^r = 0, \end{cases} \tag{5}$$

where $\bar{m}_q^r$ is the availability state seen by the policy. Thus, a missing role is either temporarily replaced by a predicted observation or left explicitly missing, in which case the policy falls back to the behavior learned in Section 4.1.

Optical-flow prediction. The prediction mechanism depends on the visual role that becomes unavailable. For wrist interruption, and for all-vision interruption where no surviving camera is available to validate a generative rollout, we use short-horizon optical-flow extrapolation. From the last two available frames, we estimate

$$F^r = \mathrm{Flow}\Big(I_{q_f-2}^r, I_{q_f-1}^r\Big), \qquad \hat{I}_q^r = \mathrm{Warp}\Big(I_{q_f-1}^r, F^r, q - q_f + 1\Big), \quad q \geq q_f, \tag{6}$$

where Flow is Farnebäck optical flow (Farnebäck, 2003) and Warp denotes short-horizon extrapolation from the last observed frame. This route is appropriate for wrist views, whose short-term visual dynamics are dominated by camera motion, and provides a conservative prediction when all visual streams are lost.

World-model prediction. Third-person interruption requires modeling changes in both robot and object configuration that cannot be captured reliably by warping the last frame. We therefore use an action-conditioned world model. At each policy query q, the model predicts the missing third-person views together with an auxiliary wrist view:

$$(\hat{I}_q^L, \hat{I}_q^R, \hat{I}_q^w) = \mathcal{G}_\psi\Big(\mathcal{M}_{q-1}, I_{q-1}^w, a^{(q-1)}, g\Big), \tag{7}$$

where $\mathcal{M}_{q-1}$ is the model's rollout memory, initialized at the interruption from the last available observations and updated with its own predictions, I_{q-1}^w is the surviving wrist observation, $a^{(q-1)}$ is the committed action chunk, and g is the task instruction. We instantiate $\mathcal{G}_\psi$ with PersistWorld (Bardhan et al., 2026), adapted to the task domain with a low-rank fine-tune.

Only $\hat{I}_q^L$ and $\hat{I}_q^R$ are supplied to the policy. The auxiliary wrist prediction $\hat{I}_q^w$ serves a different purpose: because the real wrist view remains observable, its agreement with $\hat{I}_q^w$ provides an online consistency signal for the otherwise unobserved third-person rollout. Section 4.3 describes how this signal, together with route-specific checks, determines c_q^r.

4.3 Reliability-Gated Fallback

Predicted observations are intended only as temporary substitutes for missing views. Because prediction errors can accumulate after interruption, MINT continuously monitors each prediction route using a set of route-specific reliability checks. A route remains active only while all of its checks are satisfied. Once rejected, its prediction is withdrawn for the remainder of the episode and the corresponding visual role falls back to the missing-input state learned in Section 4.1.

For a missing visual role r, let $\{z_{q,k}^r\}_{k \in \mathcal{K}_r}$ denote its reliability diagnostics at query q, where $\mathcal{K}_r$ indexes the route-specific checks. We define route reliability as

$$\rho_q^r = \prod_{k \in \mathcal{K}_r} \mathbf{1}\left[z_{q,k}^r \in \mathcal{A}_{r,k}\right], \tag{8}$$

where $\mathcal{A}_{r,k}$ denotes the admissible range of the k-th reliability diagnostic (Table 4). Thus, $\rho_q^r = 1$ only when all route-specific checks are satisfied.

The acceptance state introduced in Eq. 5 is updated as

$$c_q^r = c_{q-1}^r \wedge \rho_q^r, \qquad c_{q_f}^r = 1. \tag{9}$$

This recursive update makes rejection irreversible within an episode: once $c_q^r = 0$, predicted observations for that role are no longer supplied to the policy.

Optical-flow route. The optical-flow route uses two reliability diagnostics: consistency of the propagated view and accumulated robot motion. Both must remain within their respective admissible ranges in Eq. 8. If either check fails, $\rho_q^r = 0$, causing the route to be rejected through Eq. 9.

World-model route. The world-model route uses four reliability diagnostics: wrist consistency, visual change, model-distribution deviation, and rollout horizon. Wrist consistency compares an auxiliary predicted wrist observation $\hat{I}_q^w$ with the corresponding real wrist observation I_q^w:

$$d_q^w = D\left(\hat{I}_q^w,\, I_q^w\right), \tag{10}$$

where $D(\cdot,\cdot)$ measures normalized visual disagreement. The resulting d_q^w, together with the other three diagnostics, forms the set of route-specific checks in Eq. 8. If any diagnostic falls outside its admissible range, $\rho_q^r = 0$ and the route is rejected through Eq. 9.

Once a prediction route is withdrawn, control continues through the policy's learned missing-input capability rather than relying on continued visual reconstruction.

5 Experiments

We evaluate MINT across two VLA backbones on MAIL-Bench, use matched controls to disentangle the contributions of learning with missing visual inputs and selective visual prediction, and further examine closed-loop deployment on a physical AgiBot-G2 robot.

5.1 Experimental Setup

Models. We evaluate the RoboCasa365 Human300-trained checkpoints of $\pi_{0.5}$ (Black et al., 2025a) and GR00T N1.5 (NVIDIA et al., 2025), together with their respective MINT variants. The base models use their original policy implementations, and MINT is applied without modifying the underlying VLA architecture. All policies are evaluated on the frozen MAIL-Bench suite of 18 Atomic-Seen tasks and 900 scenes described in Section 3. We divide 18 tasks into 3 categories, which are 5 *pick-and-place* tasks, 6 *small-appliance* tasks, and 7 *fixture and navigation* tasks. We report the results of each category in Table 2. The complete task list and results are reported in Appendix C.

Training. Both MINT policies are continued for *30k* steps using the original action objective and optimization setup of their original checkpoints. Training uses a temporal availability curriculum over healthy, wrist-missing, third-person-missing and all-vision-missing observations with probabilities $\{0.60, 0.17, 0.17, 0.06\}$. For fault episodes, persistent interruptions are sampled with probability 0.7, with onset uniformly drawn from $\{0.3, 0.7\}$ of the episode. The remaining cases contain short intermittent interruptions. Unavailable visual tokens are masked from policy computation rather than replaced by informative image content.

Ablation Setup. For GR00T N1.5, we construct two matched variants to isolate the two main components of MINT. **(1)** *Clean FT + prediction* is fine-tuned for the same 30k steps using the same training data as MINT, but without missing-view examples during training. At inference time, it uses the same prediction mechanism as MINT. **(2)** *Missing-view FT* uses the same missing-view-trained checkpoint as MINT and the same serving cadence, but disables generated observations so that interrupted views remain missing. Thus, *Clean FT + prediction* versus MINT isolates the contribution of missing-view training under matched inference, while *Missing-view FT* versus MINT isolates the contribution of selective visual prediction with the policy weights held fixed. More implementation details are provided in Appendix B.

5.2 Benchmark Results

Overall Results. As demonstrated in Table 2, MINT improves the performance of VLA base model even with healthy observations. On GR00T N1.5, for example, healthy success on the *small-appliance* tasks increases from 0.290 to 0.450. Under healthy observation setting, we only have model with availability finetuning and there is no visual generation process for the missing view. This suggests that learning under partial visual observations does not introduce a trade-off with normal execution and may reduce over-reliance on view-specific superficial cues. Once visual inputs are

Table 2: **MAIL-Bench results.** Healthy denotes success rate with all cameras available, while W/T/B report scores under wrist, third-person, and all-vision interruption at 30/45/60% of completion time. AVG is the MAIL-Bench score and Δ its improvement over the corresponding base checkpoint. **Bold** marks the best result within each backbone and task category.

Method	Tasks	Healthy	Wrist missing			Third-person missing			All vision missing			AVG	Δ
			W30	W45	W60	T30	T45	T60	B30	B45	B60		
GR00T N1.5 (NVIDIA et al., 2025)													
base model	pick-and-place	**0.732**	0.163	0.276	0.341	0.165	0.224	0.424	0.000	0.000	0.010	0.234	-
	small appliances	0.290	0.153	0.277	0.398	0.245	0.296	0.403	0.110	0.170	0.226	0.257	-
	fixtures & navigation	0.386	0.178	0.253	0.385	0.308	0.319	0.398	0.029	0.084	0.119	0.246	-
+MINT	pick-and-place	0.724	**0.423**	**0.561**	**0.673**	**0.622**	**0.674**	**0.647**	**0.053**	**0.091**	**0.166**	**0.463**	+0.229
	small appliances	**0.450**	**0.493**	**0.536**	**0.631**	**0.602**	**0.628**	**0.573**	**0.307**	**0.419**	**0.528**	**0.517**	+0.260
	fixtures & navigation	**0.497**	**0.450**	**0.507**	**0.649**	**0.335**	**0.488**	**0.566**	**0.076**	**0.197**	**0.378**	**0.414**	+0.168
$\pi_{0.5}$ (Black et al., 2025a)													
base model	pick-and-place	0.636	**0.059**	0.098	0.200	0.239	0.245	0.375	0.000	0.014	0.089	0.196	-
	small appliances	0.257	0.270	0.264	0.342	0.291	0.343	0.413	0.065	0.095	0.156	0.250	-
	fixtures & navigation	0.360	**0.151**	**0.266**	0.355	**0.453**	0.469	**0.602**	0.043	**0.129**	0.254	0.308	-
+MINT	pick-and-place	**0.688**	0.027	**0.111**	**0.342**	**0.574**	**0.735**	**0.713**	**0.025**	**0.090**	**0.196**	**0.350**	+0.154
	small appliances	**0.333**	**0.291**	**0.426**	**0.350**	**0.513**	**0.372**	**0.480**	**0.218**	**0.337**	**0.370**	**0.369**	+0.119
	fixtures & navigation	**0.414**	0.131	0.261	**0.528**	0.428	**0.498**	0.546	**0.093**	0.128	**0.278**	**0.331**	+0.023

interrupted, we observe more clear improvement. Across 9 interruption conditions, MINT outperforms the GR00T N1.5 and $\pi_{0.5}$ base model in almost all cases. The gain spans wrist, third-person, and all-vision loss and persists across early and late interruption onsets, indicating that the benefit is not tied to a particular failure pattern. The consistent improvement across two VLA backbones (*i.e.*, GR00T N1.5 and $\pi_{0.5}$) further suggests that MINT is not specific to a single policy architecture.

Ablation Study. MINT consists of two stages: missing-view training and selective visual prediction at inference and we ablate each on GR00T N1.5. Removing either stage degrades score, and Fig. 4 shows that the two stages cover different failure regimes. Missing-view training governs how far the policy falls when a view is lost early: without it, scores at the 30% onset drop to 0.29/0.32 under wrist/third-person loss, whereas late interruptions (W60) are largely recovered by prediction alone. Prediction, in turn, determines how much of the lost view can be recovered when another view remains: it is most effective under early third-person interruption (+0.105 at T30), where the global view can still be inferred from the wrist camera, and its gain shrinks toward the 60% onset as less of the task remains. Under all-vision loss the two curves coincide, since the only prediction available there is short-horizon flow extrapolation of the last frames.

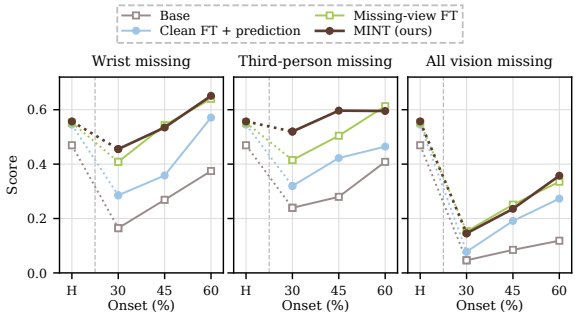

Figure 4: **Ablation on GR00T N1.5.** Scores averaged over the three task categories at each interruption onset. "H" marks the healthy-input score of each variant.

5.3 Real Robot Experiment

The simulation experiments test robustness at scale; we next ask whether the same observation-interface design transfers to a physical robot without changing the underlying VLA architecture. We deploy MINT on an AgiBot G2 using an availability-aware GR00T N1.5 policy fine-tuned on 203 successful teleoperated episodes spanning four tabletop manipulation tasks. The G2 head camera provides the third-person view and the right wrist camera the wrist view. On all four tasks, we compare GR00T N1.5 with GR00T N1.5+MINT under healthy execution and under sustained third-person, wrist, and all-camera interruption (Figure 6).

Figure 5 shows a representative closed-loop drawer episode in which the third-person camera is interrupted mid-episode and the world-model route is activated; the healthy reference episode is shown in Appendix D (Figure 7). After interruption the policy no longer receives real third-person

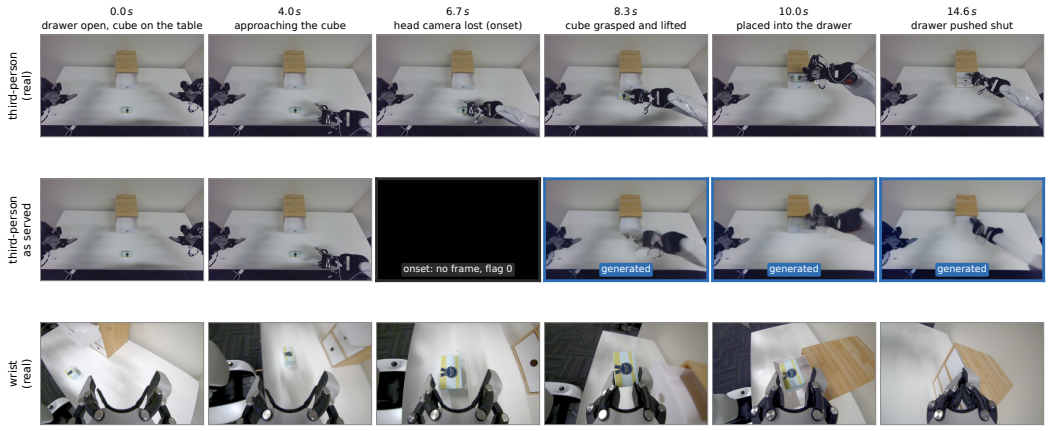

Figure 5: Episode on 'Cube to Drawer' task with the third-person camera interrupted at step 50. The policy completes grasping, transport, release and drawer closure. **Top:** real third-person stream, hidden from the policy after interruption. **Middle:** the third-person view stream and prediction by the world model. **Bottom:** the wrist stream, which stays available.

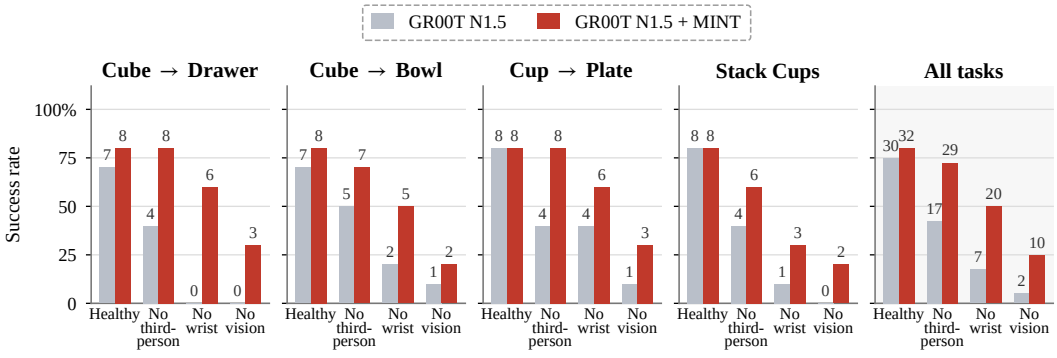

Figure 6: **Real robot experiment on the AgiBot G2.** Successful rate of GR00T N1.5 and MINT over 10 trials per task and condition. Numbers above the bars are successful trials.

camera frames. From the next query the world model supplies the missing view and the robot completes the task.

6 Conclusion

The loss of a visual modality is not simply a corruption of its content but the absence of information, and robust policies should treat the two differently. MAIL-Bench evaluates this failure mode directly and shows that camera role matters: losing a single wrist view can be more disruptive than losing both third-person views. MINT addresses the problem by separating robustness to missingness from visual prediction: availability-aware fine-tuning teaches the policy to continue acting without a view, while the router supplies predicted observations only while they remain reliable. Across two VLA backbones, this design improves aggregate interruption robustness without modifying the underlying policy architecture, and the same observation interface transfers to real-robot deployment. More broadly, our results suggest that prediction should complement, rather than replace, a policy's ability to act when information is unavailable.

Limitations and Future Work. The current release of MAIL-Bench is evaluated on RoboCasa365 with one embodiment and 18 Atomic-Seen tasks. Although the protocol is readily extensible to other multi-camera benchmarks and embodiments, broader empirical validation remains future work. MINT's world-model route also incurs additional inference and memory overhead, motivating more efficient prediction mechanisms.

REFERENCES

Jai Bardhan, Patrik Drozdik, Josef Sivic, and Vladimir Petrik. Persistent robot world models: Stabilizing multi-step rollouts via reinforcement learning. In *European Conference on Computer Vision (ECCV)*, 2026. arXiv:2603.25685.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *Proceedings of the 9th Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 2025a.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. In *Robotics: Science and Systems (RSS)*, 2025b. doi: 10.15607/RSS.2025.XXI.010.

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

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan-ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. In *International Conference on Machine Learning (ICML)*, 2026.

Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Image Analysis (SCIA)*, volume 2749 of *Lecture Notes in Computer Science*, pp. 363–370. Springer, 2003. doi: 10.1007/3-540-45103-X_50.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. LIBERO-Plus: A progressive robustness benchmark for visual-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 38574–38583, 2026.

Yanjiang Guo, Lucy Xiaoyang Shi, Jianyu Chen, and Chelsea Finn. Ctrl-World: A controllable generative world model for robot manipulation. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2510.10125.

Asher J. Hancock, Allen Z. Ren, and Anirudha Majumdar. Run-time observation interventions make vision-language-action models more visually robust. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9499–9506, 2025. doi: 10.1109/ICRA55743.2025.11128017.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations (ICLR)*, 2019.

Hassan Ismkhan and Hamid Bouchahcia. Reinforcement learning-guided retrieval with soft fusion for robust multimodal imitation learning under missing modalities. *arXiv preprint arXiv:2606.15514*, 2026.

Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. RLBench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters (RA-L)*, 5 (2):3019–3026, 2020. doi: 10.1109/LRA.2020.2974707.

Kyungwon Kim and Dosik Hwang. MUST: Modality-specific representation-aware transformer for diffusion-enhanced survival prediction with missing modality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 30312–30321, 2026.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P. Foster, Pannag R. Sanketi, Quan Vuong, et al. OpenVLA: An open-source vision-language-action model. In *Proceedings of the 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025.

Myungkyu Koo, Daewon Choi, Taeyoung Kim, Kyungmin Lee, Changyeon Kim, Younggyo Seo, and Jinwoo Shin. HAMLET: Switch your vision-language-action model into a history-aware policy. In *International Conference on Learning Representations (ICLR)*, 2026.

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, et al. BEHAVIOR-1K: A benchmark for embodied AI with 1,000 everyday activities and realistic simulation. In *Proceedings of the 6th Conference on Robot Learning (CoRL)*, volume 205 of *Proceedings of Machine Learning Research*, pp. 80–93. PMLR, 2023.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Oier Mees, Karl Pertsch, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. In *Proceedings of the 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pp. 3705–3728. PMLR, 2025.

Zijun Lin, Jiafei Duan, Haoquan Fang, Dieter Fox, Ranjay Krishna, Cheston Tan, and Bihan Wen. FailSafe: Reasoning and recovery from failures in vision-language-action models. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2026. arXiv:2510.01642.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023) Datasets and Benchmarks Track*, volume 36, 2023.

Chenyang Ma, Kai Lu, Guangyu Yang, Jiuming Liu, Shitong Xu, Bill Byrne, Ioannis Havoutis, Niki Trigoni, and Andrew Markham. CycleVLA: Proactive self-correcting vision-language-action models via subtask backtracking and minimum Bayes risk decoding. *arXiv preprint arXiv:2601.02295*, 2026.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):7327–7334, 2022. doi: 10.1109/LRA.2022.3180108.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-scale simulation of household tasks for generalist robots. In *Robotics: Science and Systems (RSS)*, 2024. doi: 10.15607/RSS.2024.XX.050.

Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. RoboCasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *International Conference on Learning Representations (ICLR)*, 2026.

NVIDIA. NVIDIA Isaac GR00T: A foundation model for generalist robots. `https://github.com/NVIDIA/Isaac-GR00T`, 2025.

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, et al. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. In *Robotics: Science and Systems (RSS)*, 2024. doi: 10.15607/RSS.2024.XX.090.

Physical Intelligence. openpi: Open-source models and packages for robotics. https://github.com/Physical-Intelligence/openpi, 2025.

Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. THE COLOSSEUM: A benchmark for evaluating generalization for robotic manipulation. In *Robotics: Science and Systems (RSS)*, 2024. doi: 10.15607/RSS.2024.XX.133.

Skand Skand, Bikram Pandit, Chanho Kim, Li Fuxin, and Stefan Lee. Simple masked training strategies yield control policies that are robust to sensor failure. In *Proceedings of the 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4463–4482. PMLR, 2025.

Sangmim Song, Sarath Kodagoda, Marc Carmichael, and Karthick Thiyagarajan. Overcoming visual clutter in vision language action models via concept-gated visual distillation. *arXiv preprint arXiv:2603.10340*, 2026.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-Kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Demonstrating GPU parallelized robot simulation and rendering for generalizable embodied AI with ManiSkill3. In *Robotics: Science and Systems (RSS)*, 2025. doi: 10.15607/RSS.2025.XXI.021.

Pankhuri Vanjani, Paul Mattes, Xiaogang Jia, Vedant Dave, and Rudolf Lioutikov. DisDP: Robust imitation learning via disentangled diffusion policies. *Reinforcement Learning Journal*, 6: 1180–1199, 2025.

Kevin Vogt-Lowell, Theodoros Tsiligkaridis, Rodney Lafuente-Mercado, Surabhi Ghatti, Shanghua Gao, Marinka Zitnik, and Daniela Rus. When sensors fail: Temporal sequence models for robust ppo under sensor drift. *arXiv preprint arXiv:2603.04648*, 2026. ICLR 2026 CAO Workshop.

Hu Wang, Yuanhong Chen, Congbo Ma, Jodie Avery, Louise Hull, and Gustavo Carneiro. Multi-modal learning with missing modality via shared-specific feature modelling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15878–15887, 2023. doi: 10.1109/CVPR52729.2023.01524.

Zhijie Wang, Zhehua Zhou, Jiayang Song, Yuheng Huang, Zhan Shu, and Lei Ma. VLATest: Testing and evaluating vision-language-action models for robotic manipulation. *Proceedings of the ACM on Software Engineering (FSE)*, 2(FSE):1615–1638, 2025. doi: 10.1145/3729343.

Siyu Xu, Yunke Wang, Zijian Wang, Dihao Zhu, Chenghao Xia, Chengbin Du, Daochang Liu, Tao Huang, and Chang Xu. StellaVLA: In-context structured demonstration for generalizable vision-language-action models. *arXiv preprint arXiv:2608.11671*, 2026.

Xiaomeng Xu, Dominik Bauer, and Shuran Song. RoboPanoptes: The all-seeing robot with whole-body dexterity. In *Robotics: Science and Systems (RSS)*, 2025a. doi: 10.15607/RSS.2025.XXI.042.

Yingxue Xu, Fengtao Zhou, Chenyu Zhao, Yihui Wang, Can Yang, and Hao Chen. Distilled prompt learning for incomplete multimodal survival prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5102–5111, 2025b. doi: 10.1109/CVPR52734.2025.00481.

Bowen Yan, Jiahao Xiao, Kehui Liu, Jianbo Zhang, Zicheng Zhang, Qi Jia, Zhongjie Jia, Haoming Song, Chunyi Li, Bin Zhao, and Guangtao Zhai. Can vlms diagnose and recover from vla manipulation faults? In *International Conference on Machine Learning (ICML)*, 2026.

Zhanguang Zhang, Zhiyuan Li, Behnam Rahmati, Rui Heng Yang, Yintao Ma, Amir Rasouli, Sajjad Pakdamansavoji, Yangzheng Wu, Lingfeng Zhang, Tongtong Cao, Feng Wen, Xinyu Wang, Xingyue Quan, and Yingxue Zhang. Do world action models generalize better than VLAs? A robustness study. *arXiv preprint arXiv:2603.22078*, 2026.

Reproducibility statement

All reported results are generated from frozen evaluation artifacts using committed scripts; no table or figure is edited manually. MAIL-Bench is released with the evaluation kernel, frozen benchmark suite and scene bank, scorer, reference servers, and scripts used to reproduce the reported results.

A run is considered valid only if its certificates verify complete healthy and interruption coverage, consistent policy configuration, and no audit failures. Each run records the benchmark version, scene and simulator-state identifiers, random seed, interruption onset, healthy-reference digest, checkpoint hash, serving configuration, and per-episode outcome (Appendix **??**).

All MINT constants were fixed on development scenes and seeds disjoint from the frozen MAIL-Bench suite before benchmark evaluation and were not tuned on benchmark outcomes (Appendix B.3). The scorer JSON, table and figure generators, and registered configuration files are included in the project repository.

Ethics statement

The simulation experiments use public assets and checkpoints and involve no human participants or private data. The benchmark follows RoboCasa's non-commercial academic licence. Because generated observations may temporarily conceal the loss of a physical sensor, MINT should not be interpreted as a safety mechanism or certification. Real-world deployment requires independent sensor-failure detection, hardware-specific validation of the reliability gates, and an appropriate fail-safe control policy.

Use of large language models

Large language models were used for language editing, manuscript organization, and assistance with writing and debugging analysis and visualization code. They were not used to generate experimental results or determine scientific conclusions. All experimental designs, executions, numerical claims, and citations were independently verified by the authors against source code, registered experimental records, and raw result files.

A MAIL-BENCH DETAILS

Suite and horizons. The suite fixes the 18 Atomic-Seen tasks, 50 scenes per task and their horizons (Table 3) under RoboCasa 1.0.1.

Table 3: **Task suite.** The 18 RoboCasa365 Atomic-Seen tasks and their horizons in control steps (20 Hz).

Task	Horizon	Task	Horizon
CloseBlenderLid	900	PickPlaceCounterToStove	600
CloseFridge	900	PickPlaceDrawerToCounter	750
CloseToasterOvenDoor	450	PickPlaceSinkToCounter	900
CoffeeSetupMug	600	PickPlaceToasterToCounter	600
NavigateKitchen	450	SlideDishwasherRack	450
OpenCabinet	1050	TurnOffStove	750
OpenDrawer	750	TurnOnElectricKettle	450
OpenStandMixerHead	450	TurnOnMicrowave	450
PickPlaceCounterToCabinet	750	TurnOnSinkFaucet	600

Base policies. We evaluate the RoboCasa365 Human300-trained checkpoints of $\pi_{0.5}$ (Black et al., 2025a) and GR00T N1.5 (NVIDIA et al., 2025), served through their native runtimes, openpi (Physical Intelligence, 2025) and Isaac-GR00T (NVIDIA, 2025). Under healthy observations, $\pi_{0.5}$ solves 362 of the 900 scenes ($H = 0.402$; 0.396 on the RoboCasa365 leaderboard) and GR00T N1.5 solves 405 ($H = 0.450$; 0.430 in the original paper and 0.507 on the leaderboard, re-evaluated under the 1.5× RoboCasa 1.0.1 horizons that MAIL-Bench pins). Missing-view training does not reduce healthy success: on GR00T N1.5, *Clean FT + prediction*, *Missing-view FT* and MINT reach $H = 0.532$, 0.540 and 0.544 (Table 6).

Auxiliary fault operators. The released kernel also implements blackout, freeze, stale delivery, burst dropout, finite-duration loss and recovery for diagnostic use; these are not part of the MAIL-Bench official score.

Determinism. Policy sampling is seeded from the benchmark seed and the simulator is reset to the frozen scene state before every rollout, so paired rollouts differ only through the camera intervention, up to small GPU-renderer differences that MAIL-Bench records.

B MINT IMPLEMENTATION DETAILS

B.1 MISSING-VIEW TRAINING

We describe the GR00T N1.5 implementation; $\pi_{0.5}$ follows the same recipe within its native backbone. Training uses the curriculum $\mathcal{C}$ of Eq. 4 with the values in Table 4: a faulted episode loses a camera group either persistently from a random onset or in short intermittent bursts, and the two third-person cameras always share one availability state.

Unavailable cameras are represented exactly as in the main method: their availability flag is 0, their image slot contains an inert zero placeholder, and their visual tokens are excluded from policy computation through the attention mask

$$\mathrm{Attn}(Q,K,V) = \mathrm{softmax}\Big(\tfrac{QK^\top}{\sqrt{d}} + \log M\Big)V, \quad M_{ij} = \begin{cases} m^{\,r(j)} & j \text{ a visual token of group } r(j), \\ 1 & \text{otherwise,} \end{cases} \tag{11}$$

with $\log 0 = -\infty$, so an unavailable camera contributes zero attention weight while an available camera is processed as usual; changing the pixels of a masked camera leaves the actions unchanged.

B.2 SERVING AND PREDICTION ROUTES

Withdrawal. Withdrawal follows Eq. 9: once a route fails a check, the role returns to the missing input used in training for the rest of the episode. The optical-flow route is checked only after a

minimum tenancy and has no upper limit on its duration; its propagated-view consistency is a flow confidence, the product of a photometric-consistency term, an accumulated-drift term and the fraction of pixels with a valid warp source. The world-model route checks every generated block from the first and is withdrawn at the first rejected block or at the generation cap. The two third-person cameras are admitted and withdrawn together.

World-model route. We use PersistWorld because it is action-conditioned and sustains multi-step rollouts within the serving budget, and adapt it to the 18 tasks with a low-rank fine-tune on the demonstrations. On held-out demonstrations, adaptation raises reconstruction PSNR from 16.4 to 18.9 dB and lowers the fraction of out-of-domain conditioning states from 0.18 to 0.01. The rollout memory $\mathcal{M}$ of Eq. 7 is initialized from the last available observations, and the auxiliary wrist prediction is used only for the consistency check. The same adaptation is used on the AgiBot G2 (Appendix D).

B.3 Choice of Hyperparameters

All hyperparameters were fixed on development scenes disjoint from the 900 benchmark scenes, before benchmark evaluation (Table 4).

Table 4: **Hyperparameters of MINT.**

Hyperparameter	Value
Optical-flow route	
Min. flow confidence (wrist / third-person)	0.05 / 0.08
Motion budget, wrist (end-effector + base)	0.15 m / 0.70 rad
Motion budget, third-person (base)	0.30 m / 0.70 rad
Minimum tenancy	64 steps
World-model route	
Max. wrist consistency error d_q^w	0.20
Visual change (max / min while moving)	0.35 / 0.002
Max. out-of-domain conditioning fraction	0.35
Generation cap	8 blocks (64 steps)
Training and serving	
Curriculum $\mathcal{C}$ (healthy / wrist / third-person / all vision)	0.60 / 0.17 / 0.17 / 0.06
Persistent-interruption probability	0.7
Persistent onset (fraction of episode)	$\mathcal{U}[0.3, 0.7]$
Fine-tuning steps	30k
Query interval	8 steps
Action horizon h (GR00T N1.5 / $\pi_{0.5}$)	16 / 50

C MAIL-Bench per-task and per-condition results

All tables are generated from the scorer output of runs with complete healthy and fault coverage and no audit failure; Table 7 gives paired bootstrap intervals. The task categories of Table 2 are pick-and-place (the five PickPlace tasks), small appliances (CloseBlenderLid, CoffeeSetupMug, OpenStandMixerHead, SlideDishwasherRack, TurnOnElectricKettle, TurnOnMicrowave) and fixtures and navigation (CloseFridge, CloseToasterOvenDoor, NavigateKitchen, OpenCabinet, OpenDrawer, TurnOffStove, TurnOnSinkFaucet). In the per-task tables, N_j^H is the number of healthy successes out of 50 scenes and "n/a" marks a task without healthy success, which contributes zero to S_j; values with small N_j^H rest on few scenes.

C.1 Compute cost

All experiments run on NVIDIA GeForce RTX 4090 GPUs (24 GB). Evaluating one arm on MAIL-Bench takes 60–70 single-worker hours, and fine-tuning takes 12.9 GPU-hours for GR00T N1.5 (30k steps, one GPU) and 15.1 GPU-hours for $\pi_{0.5}$ (30k steps, two GPUs). A policy query costs about 5.0 TFLOPs for $\pi_{0.5}$ and 2.5 TFLOPs for GR00T N1.5. MINT does not increase the per-step wall-clock time of healthy execution; under interruption, the optical-flow route runs on the CPU,

Table 5: **Main results.** Healthy success H, conditional scores M_c under wrist (W), third-person (T) and all-vision (B) interruption at the 30/45/60% onsets, and S_{MAIL} (Eq. 2), averaged over the 18 tasks.

		Wrist missing			Third-person missing			All vision missing			
Policy	H	W30	W45	W60	T30	T45	T60	B30	B45	B60	S_{MAIL}
$\pi_{0.5}$	0.402	0.165	0.219	0.307	0.340	0.365	0.476	0.038	0.086	0.176	**0.257**
$\pi_{0.5}$ + MINT	0.463	0.156	0.274	0.417	0.497	0.522	0.571	0.116	0.187	0.286	**0.349**
GR00T N1.5	0.450	0.165	0.267	0.377	0.247	0.285	0.407	0.048	0.089	0.124	**0.246**
GR00T N1.5 + MINT	0.544	0.457	0.532	0.650	0.504	0.586	0.591	0.147	0.242	0.369	**0.462**

Table 6: **GR00T N1.5 ablation.** All variants except the base are served at the same eight-step cadence. *Clean FT + prediction* is fine-tuned on the same data and steps as MINT without missing-view examples; *Missing-view FT* is the MINT checkpoint without generated views. Δ is S_{MAIL} minus the row above.

		Wrist missing			Third-person missing			All vision missing				
Arm	H	W30	W45	W60	T30	T45	T60	B30	B45	B60	S_{MAIL}	Δ
Base	0.450	0.165	0.267	0.377	0.247	0.285	0.407	0.048	0.089	0.124	**0.246**	
Clean FT + prediction	0.532	0.294	0.369	0.580	0.316	0.419	0.455	0.079	0.191	0.279	**0.351**	+0.105
Missing-view FT	0.540	0.412	0.542	0.643	0.405	0.495	0.605	0.151	0.252	0.334	**0.438**	+0.086
MINT	0.544	0.457	0.532	0.650	0.504	0.586	0.591	0.147	0.242	0.369	**0.462**	+0.024

Table 7: **Bootstrap 95% intervals.** Scenes are resampled within each task (10,000 resamples), and differences between arms are paired on the same resampled scenes. Top: S_{MAIL} and paired differences. Bottom: per-condition paired differences between each backbone with MINT and its base.

Arm	S_{MAIL}	95% interval	Paired difference	ΔS_{MAIL}	95% interval
$\pi_{0.5}$	0.257	[0.221, 0.275]	$\pi_{0.5}$ + MINT $-$ $\pi_{0.5}$	$+0.091$	$[+0.063, +0.129]$
$\pi_{0.5}$ + MINT	0.349	[0.325, 0.365]	GR00T N1.5 + MINT $-$ GR00T N1.5	$+0.216$	$[+0.190, +0.240]$
GR00T N1.5	0.246	[0.227, 0.265]	Clean FT + prediction $-$ GR00T N1.5	$+0.105$	$[+0.079, +0.130]$
Clean FT + prediction	0.351	[0.332, 0.369]	Missing-view FT $-$ Clean FT + prediction	$+0.086$	$[+0.061, +0.109]$
Missing-view FT	0.438	[0.417, 0.455]	GR00T N1.5 + MINT $-$ Missing-view FT	$+0.024$	$[+0.008, +0.044]$
GR00T N1.5 + MINT	0.462	[0.443, 0.479]	GR00T N1.5 + MINT $-$ Clean FT + prediction	$+0.111$	$[+0.087, +0.133]$

	$\pi_{0.5}$ + MINT $-$ $\pi_{0.5}$		GR00T N1.5 + MINT $-$ GR00T N1.5	
Condition	ΔM_c	95% interval	ΔM_c	95% interval
W30	-0.010	$[-0.068, +0.044]$	$+0.291$	$[+0.235, +0.345]$
W45	$+0.056$	$[-0.022, +0.145]$	$+0.264$	$[+0.206, +0.321]$
W60	$+0.110$	$[+0.018, +0.180]$	$+0.273$	$[+0.208, +0.344]$
T30	$+0.157$	$[+0.079, +0.237]$	$+0.257$	$[+0.179, +0.331]$
T45	$+0.158$	$[+0.095, +0.226]$	$+0.301$	$[+0.223, +0.372]$
T60	$+0.095$	$[+0.043, +0.175]$	$+0.184$	$[+0.114, +0.249]$
B30	$+0.078$	$[+0.042, +0.113]$	$+0.099$	$[+0.067, +0.131]$
B45	$+0.101$	$[+0.046, +0.159]$	$+0.152$	$[+0.105, +0.196]$
B60	$+0.110$	$[+0.065, +0.185]$	$+0.245$	$[+0.198, +0.292]$

whereas the world model adds 9.2 GiB of GPU memory and about 100 TFLOPs per generation, with 7.35 (GR00T N1.5) and 7.72 ($\pi_{0.5}$) generations per third-person-missing cell on average.

Table 8: Per-task, per-condition results for $\pi_{0.5}$.

Task	N_j^H	W30	W45	W60	T30	T45	T60	B30	B45	B60	S_j
CloseBlenderLid	0	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	0.000
CloseFridge	23	0.70	0.57	0.74	0.09	0.17	0.39	0.09	0.26	0.57	0.403
CloseToasterOvenDoor	2	0.00	1.00	1.00	0.50	0.50	1.00	0.00	0.50	1.00	0.554
CoffeeSetupMug	5	0.00	0.00	0.00	0.20	0.20	0.20	0.00	0.00	0.00	0.070
NavigateKitchen	0	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	0.000
OpenCabinet	34	0.03	0.12	0.18	0.09	0.15	0.15	0.00	0.00	0.00	0.139
OpenDrawer	28	0.00	0.04	0.18	0.61	0.75	0.82	0.00	0.00	0.00	0.295
OpenStandMixerHead	27	0.81	0.85	0.96	0.56	0.63	0.67	0.11	0.19	0.22	0.554
PickPlaceCounterToCabinet	39	0.05	0.13	0.15	0.00	0.00	0.13	0.00	0.00	0.03	0.127
PickPlaceCounterToStove	38	0.16	0.18	0.26	0.45	0.55	0.87	0.00	0.03	0.26	0.352
PickPlaceDrawerToCounter	23	0.04	0.04	0.13	0.09	0.09	0.17	0.00	0.04	0.09	0.116
PickPlaceSinkToCounter	45	0.04	0.13	0.31	0.44	0.44	0.49	0.00	0.00	0.00	0.277
PickPlaceToasterToCounter	14	0.00	0.00	0.14	0.21	0.14	0.21	0.00	0.00	0.07	0.107
SlideDishwasherRack	32	0.50	0.50	0.62	0.53	0.69	0.84	0.12	0.16	0.56	0.517
TurnOffStove	9	0.33	0.11	0.22	0.89	0.78	0.89	0.11	0.11	0.11	0.374
TurnOnElectricKettle	13	0.31	0.23	0.46	0.46	0.54	0.77	0.15	0.23	0.15	0.357
TurnOnMicrowave	0	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	0.000
TurnOnSinkFaucet	30	0.00	0.03	0.17	1.00	0.93	0.97	0.10	0.03	0.10	0.393
Mean over 18 tasks	362	0.165	0.219	0.307	0.340	0.365	0.476	0.038	0.086	0.176	**0.257**

Table 9: Per-task, per-condition results for $\pi_{0.5}$ + MINT.

Task	N_j^H	W30	W45	W60	T30	T45	T60	B30	B45	B60	S_j
CloseBlenderLid	2	0.00	0.50	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.104
CloseFridge	23	0.70	0.74	0.78	0.48	0.48	0.70	0.26	0.22	0.70	0.550
CloseToasterOvenDoor	9	0.11	0.22	0.56	0.11	0.22	0.33	0.22	0.22	0.33	0.251
CoffeeSetupMug	13	0.00	0.00	0.00	0.46	0.38	0.69	0.00	0.00	0.15	0.195
NavigateKitchen	1	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.102
OpenCabinet	36	0.06	0.22	0.31	0.11	0.36	0.25	0.00	0.03	0.03	0.208
OpenDrawer	31	0.00	0.16	0.45	0.58	0.74	0.81	0.03	0.13	0.26	0.378
OpenStandMixerHead	37	0.38	0.62	0.86	0.70	0.65	0.78	0.59	0.78	0.86	0.698
PickPlaceCounterToCabinet	39	0.05	0.15	0.62	0.67	0.87	0.79	0.03	0.21	0.46	0.463
PickPlaceCounterToStove	36	0.00	0.17	0.44	0.69	0.94	0.67	0.00	0.06	0.14	0.383
PickPlaceDrawerToCounter	26	0.04	0.08	0.23	0.27	0.38	0.35	0.08	0.12	0.31	0.237
PickPlaceSinkToCounter	44	0.05	0.05	0.27	0.50	0.77	0.80	0.02	0.00	0.00	0.333
PickPlaceToasterToCounter	27	0.00	0.11	0.15	0.74	0.70	0.96	0.00	0.07	0.07	0.335
SlideDishwasherRack	32	0.66	0.72	0.88	0.84	0.84	0.91	0.50	0.81	0.84	0.764
TurnOffStove	8	0.00	0.38	0.25	0.88	0.88	0.88	0.00	0.00	0.25	0.366
TurnOnElectricKettle	14	0.21	0.21	0.36	0.57	0.36	0.50	0.21	0.43	0.36	0.349
TurnOnMicrowave	2	0.50	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.104
TurnOnSinkFaucet	37	0.05	0.11	0.35	0.84	0.81	0.86	0.14	0.30	0.38	0.458
Mean over 18 tasks	417	0.156	0.274	0.417	0.497	0.522	0.571	0.116	0.187	0.286	**0.349**

Table 10: Per-task, per-condition results for GR00T N1.5.

Task	N_j^H	W30	W45	W60	T30	T45	T60	B30	B45	B60	S_j
CloseBlenderLid	3	0.00	0.00	0.67	0.00	0.00	0.00	0.00	0.00	0.00	0.073
CloseFridge	35	0.57	0.71	0.74	0.17	0.31	0.37	0.20	0.29	0.46	0.453
CloseToasterOvenDoor	12	0.25	0.50	0.92	0.08	0.17	0.42	0.00	0.00	0.08	0.266
CoffeeSetupMug	5	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.030
NavigateKitchen	5	0.20	0.20	0.20	0.40	0.00	0.00	0.00	0.20	0.00	0.130
OpenCabinet	32	0.19	0.25	0.44	0.06	0.06	0.16	0.00	0.03	0.00	0.183
OpenDrawer	28	0.04	0.11	0.29	0.54	0.54	0.61	0.00	0.07	0.18	0.292
OpenStandMixerHead	26	0.31	0.58	0.77	0.27	0.42	0.69	0.19	0.23	0.31	0.429
PickPlaceCounterToCabinet	41	0.46	0.61	0.54	0.17	0.22	0.49	0.00	0.00	0.00	0.331
PickPlaceCounterToStove	41	0.02	0.20	0.41	0.12	0.22	0.46	0.00	0.00	0.02	0.228
PickPlaceDrawerToCounter	21	0.00	0.05	0.00	0.19	0.24	0.48	0.00	0.00	0.00	0.137
PickPlaceSinkToCounter	41	0.07	0.24	0.32	0.32	0.32	0.51	0.00	0.00	0.02	0.262
PickPlaceToasterToCounter	39	0.26	0.28	0.44	0.03	0.13	0.18	0.00	0.00	0.00	0.209
SlideDishwasherRack	24	0.38	0.62	0.58	0.79	0.71	0.79	0.33	0.38	0.54	0.560
TurnOffStove	5	0.00	0.00	0.00	0.40	0.60	0.40	0.00	0.00	0.00	0.150
TurnOnElectricKettle	22	0.09	0.32	0.23	0.41	0.50	0.59	0.14	0.27	0.36	0.335
TurnOnMicrowave	7	0.14	0.14	0.14	0.00	0.14	0.14	0.00	0.14	0.14	0.114
TurnOnSinkFaucet	18	0.00	0.00	0.11	0.50	0.56	0.83	0.00	0.00	0.11	0.247
Mean over 18 tasks	405	0.165	0.267	0.377	0.247	0.285	0.407	0.048	0.089	0.124	**0.246**

Table 11: Per-task, per-condition results for GR00T N1.5 + MINT.

Task	N_j^H	W30	W45	W60	T30	T45	T60	B30	B45	B60	S_j
CloseBlenderLid	2	0.00	0.00	0.00	0.50	0.50	0.00	0.00	0.00	0.00	0.104
CloseFridge	30	0.83	0.70	0.80	0.30	0.53	0.63	0.23	0.20	0.37	0.520
CloseToasterOvenDoor	19	0.26	0.37	0.63	0.37	0.26	0.37	0.00	0.05	0.32	0.301
CoffeeSetupMug	11	0.18	0.18	0.64	0.36	0.36	0.64	0.09	0.00	0.09	0.277
NavigateKitchen	14	0.43	0.57	0.50	0.00	0.07	0.14	0.00	0.00	0.07	0.207
OpenCabinet	41	0.78	0.73	0.78	0.22	0.39	0.54	0.05	0.07	0.15	0.453
OpenDrawer	36	0.28	0.58	0.81	0.64	0.75	0.81	0.08	0.31	0.78	0.575
OpenStandMixerHead	34	0.88	0.97	0.94	0.62	0.59	0.62	0.65	0.82	1.00	0.777
PickPlaceCounterToCabinet	42	0.71	0.79	0.88	0.83	0.90	0.76	0.24	0.38	0.57	0.691
PickPlaceCounterToStove	40	0.25	0.47	0.55	0.75	0.62	0.47	0.00	0.00	0.00	0.392
PickPlaceDrawerToCounter	19	0.05	0.05	0.21	0.21	0.32	0.53	0.00	0.00	0.11	0.185
PickPlaceSinkToCounter	42	0.33	0.57	0.88	0.74	0.79	0.79	0.00	0.02	0.02	0.498
PickPlaceToasterToCounter	38	0.76	0.92	0.84	0.58	0.74	0.68	0.03	0.05	0.13	0.550
SlideDishwasherRack	39	0.82	0.85	0.82	0.72	0.74	0.82	0.49	0.82	0.90	0.775
TurnOffStove	10	0.40	0.30	0.40	0.40	0.70	0.60	0.00	0.50	0.30	0.380
TurnOnElectricKettle	34	0.47	0.62	0.59	0.88	0.97	0.76	0.35	0.47	0.65	0.644
TurnOnMicrowave	15	0.60	0.60	0.80	0.53	0.60	0.60	0.27	0.40	0.53	0.523
TurnOnSinkFaucet	24	0.17	0.29	0.62	0.42	0.71	0.88	0.17	0.25	0.67	0.465
Mean over 18 tasks	490	0.457	0.532	0.650	0.504	0.586	0.591	0.147	0.242	0.369	**0.462**

D REAL-ROBOT SETUP

The AgiBot G2 has a head-mounted camera and one camera on each wrist, and the right arm manipulates. The head camera supplies the third-person group and the right wrist camera the wrist group. We collected 224 teleoperated episodes of four tabletop tasks and fine-tune an availability-aware GR00T N1.5 checkpoint on the 203 successful ones. Figures 7–10 show a healthy episode of each task.

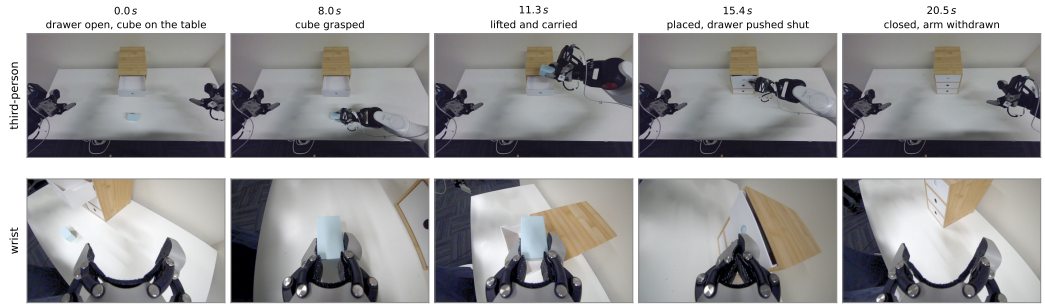

Figure 7: **Healthy drawer episode on the AgiBot G2.** Head (third-person) camera (top) and right wrist camera (bottom) at five time points.

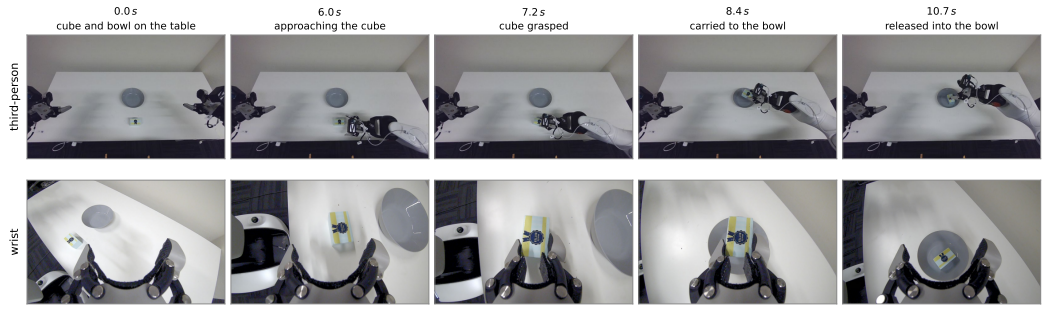

Figure 8: **Healthy cube-into-bowl episode on the AgiBot G2.** Head (third-person) camera (top) and right wrist camera (bottom) at five time points.

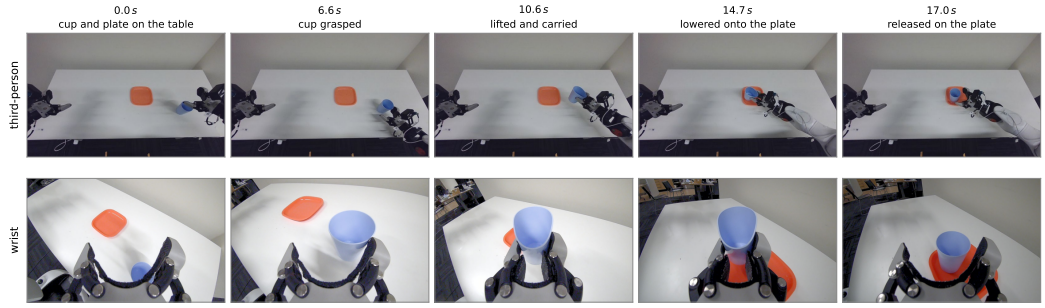

Figure 9: **Healthy cup-onto-plate episode on the AgiBot G2.** Head (third-person) camera (top) and right wrist camera (bottom) at five time points.

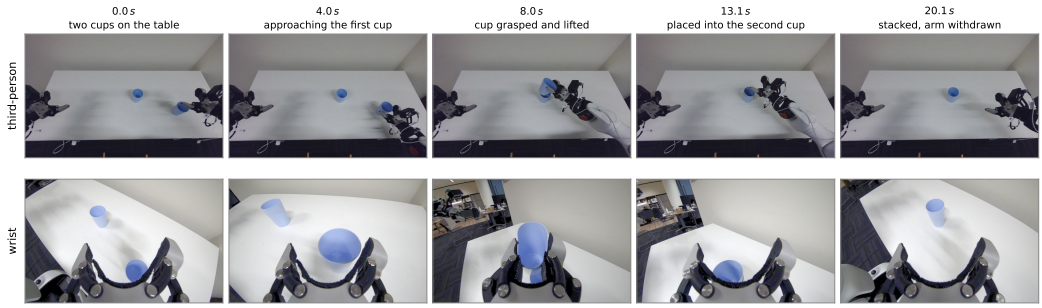

Figure 10: **Healthy stack-cups episode on the AgiBot G2.** Head (third-person) camera (top) and right wrist camera (bottom) at five time points.