IMBench: A Benchmark for Intuitive Robotic Manipulation

Anurag Maurya Sukhvansh Jain Prajwal Avhad Gautham Balachandran Ziyi Zhou
Atharva Kshirsagar Satyam Singh Bowen Li Rishabh Mukund Ritul Singh Jatin Vira
Suvonil Chatterjee Devesh K. Jha
Manav Robotics, Bengaluru, India

***Abstract*—Humans combine reasoning and motor control to solve complex manipulation tasks under diverse constraints. They build an understanding of the physical world that helps them convert reasoning into actions and quickly adapt to new scenes, tasks, and rules. We refer to this capability as intuitive manipulation. Existing benchmarks fail to capture this integration: they evaluate physical reasoning in isolation from execution, or measure policy performance without requiring explicit reasoning. We introduce IMBENCH, a benchmark designed to evaluate intuitive manipulation as an integrated capability spanning perception, physical reasoning, action generation, and iterative execution. Our tasks require models to infer task-relevant physical structure and generate feasible action sequences under explicit constraints, including contact-rich manipulation, tool use, and multi-stage dependencies. We introduce a benchmark of 35 tasks, 14K filtered trajectories, and scalable tools for generating diverse scenarios. Experiments reveal a consistent gap: vision-language models show partial physical reasoning ability but fail to produce executable plans, while state-of-the-art vision-language-action models struggle to satisfy task constraints and generalize across scenarios. These results identify intuitive manipulation as a missing axis in current foundation models and generalist robot policies, and position IMBENCH as a step toward evaluating and enabling more integrated, adaptive physical intelligence. Project Website**

I. INTRODUCTION

Recent advances in robot learning have brought the field closer to generalist robots capable of performing a broad range of tasks across diverse environments [49]. However, a key gap remains: grounding behavior in the physical world the way humans do. Humans naturally reason about object properties, relationships, environmental constraints, and dynamics, using this understanding to guide flexible, adaptive actions [38]. In contrast, current robotic systems struggle to capture and leverage such physical intuition, limiting their ability to generalize.

Consider a simple example (Fig. 1)(Top): a thin object lies flat on a table. A direct grasp is not feasible due to the geometry of the object. Humans, however, intuitively adapt, they slide the plate toward the edge of the table, enabling a stable grasp before lifting it. This behavior reflects more than passive understanding; it demonstrates the ability to convert physical understanding into goal-directed action. It involves reasoning about why a default action fails (*Understand*), mentally simulating the effects of alternate actions (*Infer*), and selecting a sequence that transforms an infeasible task into a feasible one (*Act*). Two cognitive capacities support this behavior. The first is intuitive physics: the brain's approximate mental simulation of how objects, forces, and contacts behave [50]. The second is means-end reasoning, our ability to reason about goals, intentions, and constraints [24], and to act efficiently to achieve them [40]. We refer to their integration: the conversion of physical understanding into goal-directed action, as *intuitive manipulation*.

Existing benchmarks probe the components of intuitive manipulation in isolation but rarely together. Physical-reasoning suites [6, 60] such as PhysBench [14], video-language QA benchmarks [33, 73, 68, 23, 71] like CLEVRER [79], and embodied 3D-spatial QA benchmarks [16, 46, 31, 45, 76, 75] like OpenEQA [47] evaluate whether models can answer questions about object properties, relations, scene structure, and dynamics, but they stop at inference and do not synthesize executable action. Manipulation benchmarks: LIBERO [43], RoboCasa [53], ManiSkill [52], RLBench [30], Colosseum [58], VLABench [81], and OGBench [56] evaluate policy execution across diverse scenes and objects. Intuitive manipulation poses a complementary challenge: actions grounded in physical principles on tasks which heavily require them, an axis prior benchmarks and policies are not designed to isolate. Recent vision-language-action models (VLAs) [36, 85, 65, 83] and related vision-language manipulation approaches [27, 55, 69, 80] generalize impressively across object identities and scene layouts, yet they underperform when the task structure itself requires this kind of physical reasoning, as we show empirically in §IV.

We introduce IMBENCH, a benchmark of 35 ma-

nipulation tasks built on robosuite [84] to evaluate intuitive manipulation across three capabilities: physical understanding, action proposal, and low-level execution. The tasks are intentionally non-trivial, requiring explicit physical reasoning to achieve task goals. They are organized around two key questions: (i) Can high-level planners infer task constraints from observations and propose feasible solutions? (ii) Can current physical-AI and foundation-model approaches translate physical understanding into low-level actions?

Our experiments reveal two key findings. First, vision-language models exhibit partial understanding of physical constraints. For instance, GPT-5.5 achieves $\approx 74\%$ on constraint understanding, but drops to $\approx 70\%$ on high-level planning, suggesting that the two capabilities are closely coupled. Second, performance degrades sharply when generating executable plans: even with correct high-level plans, action execution success falls to only 11%. State-of-the-art policies, including diffusion policy [13], $\pi_{0.5}$ [29], and GROOT [7], achieve low success rates on our physics-grounded tasks despite near-saturating performance on existing manipulation benchmarks such as LIBERO [43], highlighting the difficulty of IMBENCH.

a) Contributions: This paper makes three primary contributions:

- We present IMBENCH, a set of 35 robosuite manipulation tasks that test whether models understand physics and can use it to act, and we show where current foundation models and learned policies fail.
- We release a dataset and tools: ≈14K filtered trajectories, and scripts to generate more data supporting reproducible research for intuitive manipulation.
- We formalize intuitive manipulation through an *Understand–Infer–Act* decomposition and evaluate it using a three-stage VLM protocol (constraint understanding $\rightarrow$ plan proposal $\rightarrow$ execution) together with closed-loop policy rollouts, enabling systematic measurement of the understanding-to-execution gap in current foundation models and policies.

We strongly encourage reviewers to visualize the tasks using the automated Hugging Face visualizer provided in Appendix A.

II. RELATED WORK

a) Physical Reasoning and Embodied Manipulation.: Physical reasoning benchmarks study whether agents can infer object dynamics, contact interactions, and causal physical structure. PHYRE [3], I-PHYRE [42] Kinetix [48], and the Virtual Tools Game [1], as well as vision-centric suites like Physion++ [66], ComPhy [9], CRAFT [2], and CoPhy [5], alongside video-physics benchmarks [74, 57, 22, 4, 17] evaluate prediction and tool-use reasoning in simulated environments. However, these benchmarks terminate at inference, are usually MCQ-style and do not require physically executable robot behavior.

Manipulation benchmarks instead focus on policy learning and generalization across tasks and environments. LIBERO [43], CALVIN [51], RoboCasa [53], ManiSkill-HAB [62], BEHAVIOR-1K [41], FurnitureBench [25], BiGym [12], and DittoGym [26] study long-horizon manipulation, embodied interaction, and transfer across diverse scenes, often entangled with application-specific complexities as seen in household-assistance benchmarks like ALFRED [61] and AI2-THOR [37]. VIMA [32] explores multimodal prompting for robotic manipulation, while large-scale datasets such as DROID [35] and BridgeData V2 [67] emphasize scaling robot learning through diverse demonstrations. These benchmarks primarily evaluate trajectory reproduction, task completion, or policy generalization, but do not isolate whether success depends on reasoning about task-specific physical constraints. Most recently, KinDER [28] benchmarks kinematic and dynamic constraints and nonprehensile multi-object manipulation etc. IMBench is complementary with focus on reasoning in both physical and semantics axes with mixed grippers and bimanual setups, and adds hidden-state, reactive-replanning, and bimanual collaboration. KinDER also lacks force-information demonstrations. A comparison with existing benchmarks is provided in Table I.

b) Vision-Language-Action Models and Robot Planning.: Recent Vision-Language-Action (VLA) models have significantly improved robot generalization across objects, scenes, and instructions. PaLM-E [15], RT-2 [85], π-0.5 [29], VLA-R1 [78], and Gemini Robotics [64] combine large-scale multimodal pretraining with action generation and reasoning capabilities. Parallel work on language-guided planning uses large language and vision-language models for embodied decision-making and replanning [63, 70, 11, 72, 77], including recent visual-prompting techniques like MOKA [44] and PIVOT [54]. Despite strong semantic generalization and recent physically-grounded VLM perception efforts [19, 8, 10, 59], these approaches still struggle when successful execution depends on reasoning about physical feasibility, or downstream action consequences. IMBench is designed to expose this limitation directly: tasks are constructed so that physically incorrect reasoning leads to execution failure even when high-level task intent is understood. Classical task and motion planning (TAMP) methods explicitly reason

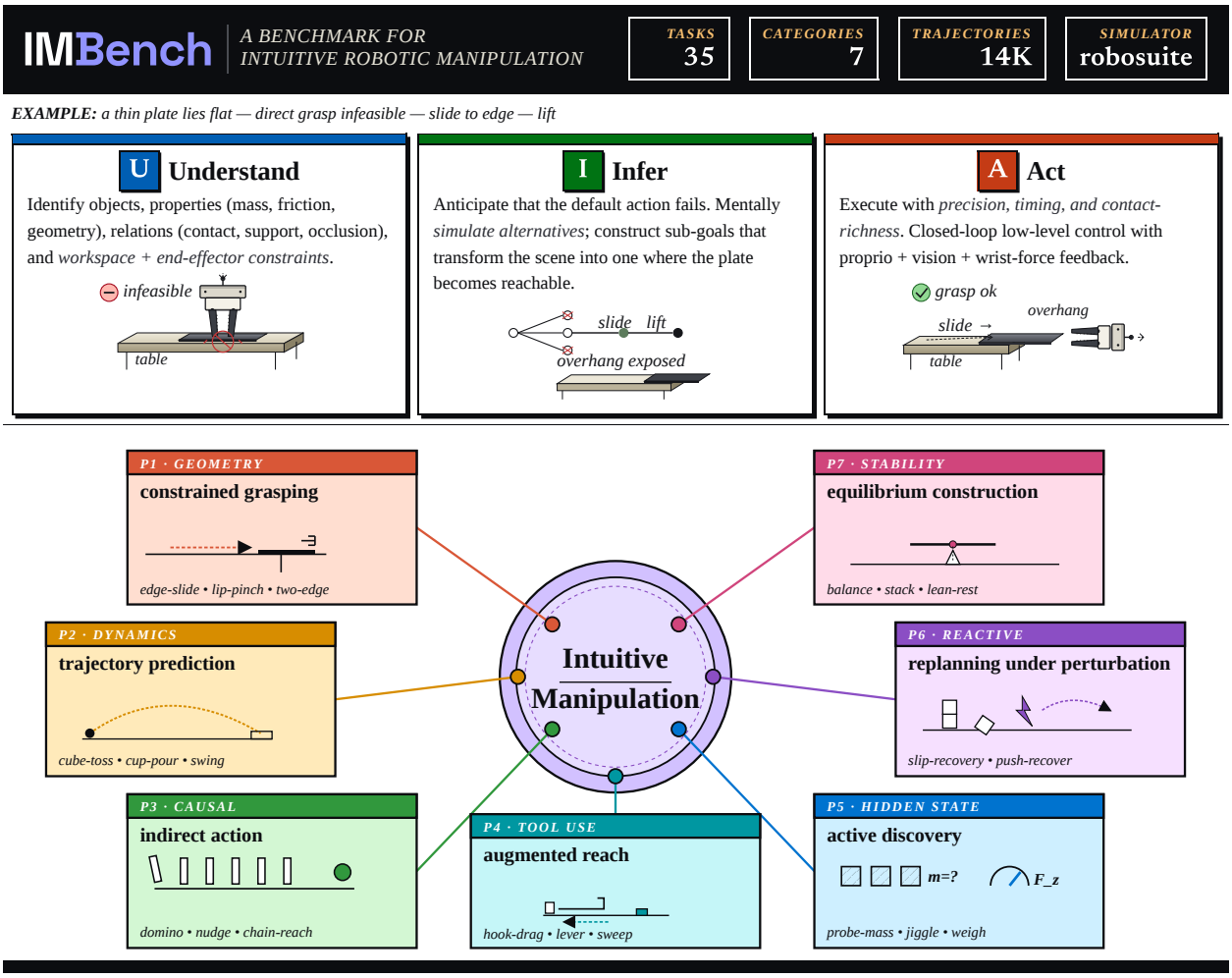


Fig. 1. **IMBench — a benchmark for intuitive manipulation. Top:** Each task is decomposed into a three-stage cognitive loop. *Understand* extracts latent physical properties from perception (e.g., a thin plate flush on a table affords no top-down grasp). *Infer* simulates forward dynamics and selects sub-goals (e.g., slide-then-grasp). *Act* executes a closed-loop interactive policy.

over symbolic and geometric constraints to ensure physically feasible manipulation [34, 21, 20, 82, 39], but typically rely on hand-engineered abstractions and environment-specific formulations. **POSITIONING:** IMBENCH complements this line of work by evaluating whether modern learned policies and foundation models acquire such physical reasoning capabilities through embodied interaction.

III. IMBENCH TASK SUITE

IMBENCH consists of 35 tasks built on `robosuite` [84]. Each task is designed around an *inference bottleneck*: a hidden fact such as infeasibility, geometry, dynamics, causality, hidden state, or stability that the agent must infer from observations to achieve the goal reliably. Successful execution needs precise closed loop physical control. Together, these tasks test intuitive physical reasoning rather than simple imitation.

Tasks follow a Gymnasium-style interface with seeded resets. Episodes terminate on success (from simulator state), failure (e.g., object leaves workspace or forbidden contact), or a time limit, with sparse rewards (1 for success, 0 otherwise). Observations include synchronized 2–4 RGB views, proprioception, gripper state, and wrist force/torque. Actions use a unified continuous space of 6-DoF end-effector deltas and gripper commands for both single-arm and bimanual setups. To ensure demonstration quality, episodes pass through a three-stage filtering pipeline: human annotators first score trajectories on a 1-5 scale (keeping only scores ≥ 4), a VLM-based agent performs anomaly and consistency checks, and a final human review finalizes the decision. More details in Appendix B8.

TABLE I

RELATED BENCHMARKS. IMBENCH IS THE ONLY BENCHMARK THAT (I) FOCUSES ON PHYSICAL REASONING, (II) EVALUATES VLM CONSTRAINT UNDERSTANDING AND PLAN GENERATION *separately* FROM CLOSED-LOOP POLICY EXECUTION, (III) COVERS PHYSICAL REASONING CATEGORIES (GEOMETRY, DYNAMICS, CAUSAL CHAINS, HIDDEN STATE, REACTIVE REPLANNING), AND (IV) PROVIDES BIMANUAL COORDINATION TASKS WITH DEMONSTRATIONS.

Benchmark	Phys. Reasoning Focus	Sep. VLM + Policy Eval	Geometry / Affordance	Dynamics / Prediction	Causal / Indirect	Hidden State	Reactive Replanning	Bimanual Coordination	Demonstrations
PHYRE [3]	✓	×	×	✓	✓	×	×	×	×
PhysBench [14]	✓	×	×	✓	×	×	×	×	×
CLEVRER [79]	✓	×	×	✓	✓	×	×	×	×
LIBERO [43]	×	×	×	×	×	×	×	×	✓
RLBench [30]	×	×	✓	×	×	×	×	×	✓
RoboCasa [53]	×	×	✓	×	×	×	×	×	✓
ManiSkill-HAB [62]	×	×	×	×	×	×	×	×	✓
VLABench [81]	×	×	×	×	✓	×	×	×	✓
Colosseum [58]	×	×	✓	×	×	×	×	×	✓
BiGym [12]	×	×	×	×	×	×	×	✓	✓
VLMgineer [18]	✓	✓	✓	×	×	×	×	×	×
KinDER [28]	✓	×	✓	✓	✓	×	×	×	✓
IMBENCH (Ours)	✓	✓	✓	✓	✓	✓	✓	✓	✓

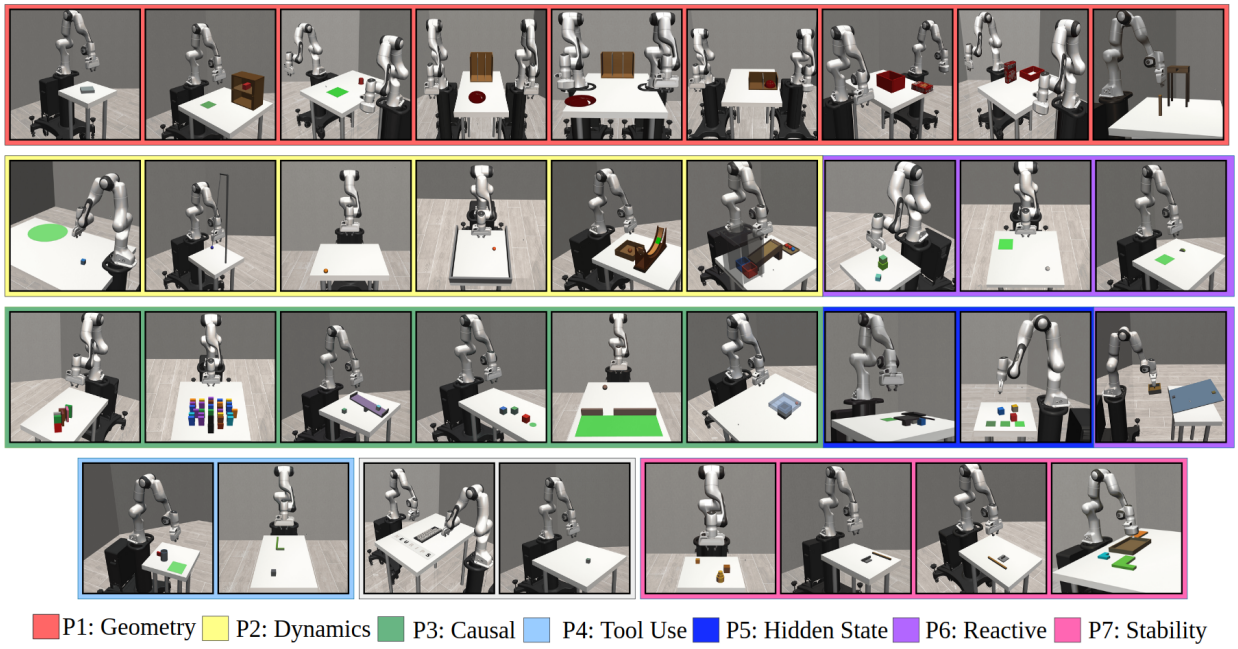


Fig. 2. **IMBench: 35 tasks across seven categories.** Each targets a distinct inference bottleneck: geometry (P1), dynamics (P2), causal chains (P3), tool use (P4), hidden state (P5), reactive replanning (P6), stability (P7); two miscellaneous tasks in grey (bottom row).

A. *Categories and inference bottlenecks*

We group the 35 tasks into seven categories, each defined by the intuition getting invoked (Fig. 2); per-task details appear in Appendix B. **P1 – Geometry and constrained grasping (9 tasks)** require the agent to make objects graspable or reason about geometry (e.g., edge-sliding thin plates, suction-to-jaw handovers, lateral rack entry), testing affordance reasoning and sub-goal ordering. **P2 – Dynamics and trajectory prediction (7 tasks)** places goals beyond static reach or moving objects (tossing, pendulum grasping, rolling-ball interception with and without bounces, color-routed sorting), probing forward simulation. **P3 – Causal and indirect action (6 tasks)** makes the target unreachable or contact-forbidden, so action must transit an intermediary chain (domino cascades with sabotaged distractors, seesaw torque balance under unknown mass, transitive pushing), probing causal-chain construction and structural reasoning. **P4 – Tool use and augmented reach (2 tasks)** requires using a tool or actions *before* the final grasp, probing multi-stage ordering under embodiment limits. **P5 –**

TABLE II

VLM EVALUATION ACROSS THREE STAGES. STAGE 1 REPORTS CONSTRAINT-UNDERSTANDING SUCCESS (%), **STAGE 2** REPORTS HIGH-LEVEL PLAN CORRECTNESS (%), AND **STAGE 3** REPORTS CLOSED-LOOP EXECUTION SUCCESS (%). A STAGE 2 PLAN IS MARKED CORRECT IF A HUMAN VERIFIES THAT THE PROPOSED SUB-GOALS SATISFY TASK CONSTRAINTS AND CAN ACHIEVE TASK SUCCESS. BEST PER COLUMN WITHIN EACH STAGE IS SHOWN IN **BOLD**.

Method	P1	P2	P3	P4	P5	P6	P7	M	Mean
Stage 1: Constraint understanding									
Gemma 4	51.0±30.6	65.2±29.6	79.3±18.5	86.0±0.0	90.0±0.0	50.6±31.1	37.5±10.1	0.0±0.0	57.8±32.0
Claude-Haiku 4.5	63.1±33.5	85.4±10.8	80.7±22.1	70.0±0.0	83.8±16.2	36.7±4.7	35.0±30.2	2.0±2.0	63.6±32.8
Claude Sonnet 4.6	65.5±33.0	**93.1**±5.0	90.0±10.0	90.0±0.0	**97.5**±2.5	62.8±27.8	65.0±27.3	0.0±0.0	**74.5**±31.2
GPT-5.4-Mini	55.2±31.0	76.7±24.3	82.2±14.3	90.0±0.0	95.0±5.0	62.2±32.5	46.7±23.9	5.0±5.0	64.8±31.6
GPT-5.5	**71.3**±35.4	75.8±30.1	**93.7**±9.0	**90.6**±4.6	94.2±4.1	**56.7**±40.3	**80.5**±15.5	**8.3**±8.3	74.1±33.0
Stage 2: Plan correctness									
Gemma 4	46.7±31.3	53.9±37.1	66.5±21.7	52.0±0.0	90.0±0.0	17.8±5.7	11.7±12.8	2.0±2.0	45.3±34.3
Claude-Haiku 4.5	56.7±36.4	72.6±27.8	67.2±26.3	24.0±0.0	83.8±16.2	48.9±9.6	28.3±25.2	2.0±2.0	55.7±33.9
Claude Sonnet 4.6	55.2±33.9	**89.4**±9.2	84.2±14.8	25.0±0.0	**97.5**±2.5	50.6±28.3	39.2±31.0	0.0±0.0	64.1±34.6
GPT-5.4-Mini	45.2±29.9	58.0±36.1	76.9±19.1	74.0±0.0	91.5±3.5	30.6±19.7	14.6±8.1	5.0±5.0	50.0±34.4
GPT-5.5	**69.0**±35.4	61.3±40.8	**93.0**±9.9	**75.9**±5.9	94.2±4.1	**56.7**±40.3	**80.0**±15.2	**8.3**±8.3	**69.5**±35.7
Stage 3: Action, uses GPT 5.5									
IMBAgent	0.0	0.0	75.0	0.0	0.0	0.0	0.0	15.0	11.3
IMBAgent-obj	0.0	0.0	**100.0**	0.0	0.0	0.0	0.0	**50.0**	**18.8**

Hidden state and active discovery (2 tasks) hides the decision-relevant variable from passive vision; identity only by displacing occluders—probing information-gathering actions and multi-modal grounding. **P6 – Reactive replanning (3 tasks)** injects mid-execution disturbances (gripper slip, knocked stack, teleporting goal), probing online failure detection and replanning. **P7 – Stability and equilibrium (4 tasks)** hinges feasibility on static balance (mixed-shape stacking, rod balancing on millimeter ridges, zero-gap L-piece packing), probing support-polygon reasoning and precision under low tolerance. **Miscellaneous (2 tasks)** adds *mirror-pick-place*, which rotates camera frames by 180° to decouple observation and action frames, and *keyboard-typing*, a bimanual task where one arm partially finishes the goal, probing partial-completion tracking and coordination. The full task list and additional details are provided in Appendix A.

IV. EXPERIMENTS

The experimental section is organized around three questions:

Q1. *Constraint understanding.* Given a scene observation(s) and a required goal, can frontier vision-language models (VLMs) correctly identify the physical constraints: infeasible grasps, hidden state, causal dependencies, that prevents naive actions from succeeding?

Q2. *High-level action proposal.* Conditioned on a (possibly correct) understanding of the scene, can the same models translate that understanding into high-level plan, expressed as a verifiable and executable sequence of sub-goals?

Q3. *Execution and generalization.* Can current visuo-motor policies execute physics grounded plans on IMBENCH and generalize beyond the training distribution?

Q1–Q2 examine *high-level* reasoning, while Q3 focuses on *low-level* execution and tests whether policies generalize along the physical factors each task targets.

A. High-level reasoning with VLMs (Q1, Q2)

We evaluate five high-level reasoners: (i) **GPT-5.5** (ii) **GPT-5.4-mini**, (iii) **Gemma-4**, (iv) **Claude Haiku-4.5**, and (v) **Claude-Sonnet-4.6**. All models use a shared chain-of-thought prompting interface to elicit structured traces for human evaluation. For each seed, we render multi-view observations and provide the task description and controller interface (DoF, gripper modes, observables). A small set of temporally spaced frames is given per task. From each response we extract two outputs: **1) Stage 1: Constraint understanding** Structured responses to probes (objects, contacts, reachability, infeasibility, hidden state, causal preconditions), evaluated via a task-specific rubric (given to the human evaluator). Success requires identifying all task-critical constraints and interactions. **2) Stage 2: High-level plan** A sequence of sub-goals $(g_1, \ldots, g_T)$ (e.g., GRASP, SLIDE, TOSS), composed by the model without hints. This tests its ability to form task-aligned plans consistent with key constraints. Restricting outputs to primitives reduces the model to MCQ style selection instead of generating new behaviors. Results are aggregated over five temperature settings. Details are provided in Appendix K.

TABLE III

POLICY SUCCESS RATES (%) ON IMBENCH, AVERAGED OVER TASKS WITHIN EACH CATEGORY. ROLLOUTS ARE OVER 20 SEEDS AND 5 ROLLOUTS EACH. BOLD MARKS THE BEST ENTRY PER ROW. ZS = ZERO-SHOT PRETRAINED CHECKPOINT; FT = FINETUNED ON TELEOPERATION DATA. DIFFUSION POLICY (DP) IS TRAINED FROM SCRATCH (FULL-TRAINING).

		$\pi_{0.5}$		**GR00T 1.5**		**DP**
		ZS	**FT**	**ZS**	**FT**	**Full-training**
P1	Geometry and constrained grasp	0.00	0.05	0.00	0.00	**0.10**
P2	Dynamics and trajectory	0.01	0.09	0.00	0.02	**0.12**
P3	Causal and indirect action	0.08	0.29	0.03	0.11	**0.37**
P4	Tool use and augmented reach	0.00	0.00	0.00	0.00	0.00
P5	Hidden state and discovery	0.00	0.08	0.00	0.00	**0.25**
P6	Reactive replanning	0.00	0.17	0.00	0.00	**0.24**
P7	Stability and equilibrium	0.00	**0.32**	0.00	0.01	0.12
	Miscellaneous	0.01	0.06	0.00	0.02	**0.65**
	Mean	0.01	0.15	0.00	0.02	**0.24**

1) Results: understanding (Stage 1): Table VIII shows that frontier VLMs describe most constraints competently, but the failures concentrate on a small, consistent set of *tasks* rather than on whole categories (Table VIII). Several tasks are understood almost perfectly by all models, including `domino-single`, `indirect-push`, `linear-intercept`, `bounce-intercept`, `pendulum-grasp`, and `sheltered-grasp` (90–100%). These strong results raise category-level scores for geometry (P1), dynamics (P2), and causal action (P3), with Sonnet 4.6 (74.5%) and GPT-5.5 (74.1%) achieving the best overall Stage-1 performance (Table II).

The weakest performance appears on tasks such as `cracker-box-jaw`, `cracker-box-suction`, `shape-stack`, and `slip-recovery`, where models recognize the objects but fail to identify the key grasp or stability constraint. Interestingly, `balance-hard` often performs better than `balance-medium` because its prompt explicitly encourages center-of-mass reasoning. Stability and equilibrium tasks (P7) remain the weakest overall category (37–80%) escept miscellaneous category. Additional analysis is provided in Appendix U.

2) Results: action proposal (Stage 2): The understanding-to-plan gap is itself task-specific. The largest drop from understanding to planning appears in tasks that require correctly ordered actions, while static geometry tasks largely retain their Stage-1 performance (Table IX). `slide-catch` is the clearest example: despite Stage-1 scores of 88–95%, most models fail completely in Stage 2 because their plans miss important timing and tracking steps. Similar failures appear in `tool-retrieve`, `cup-inversion`, `stack-collapse-recovery`, and `cup-extract`. In contrast, tasks such as `gap-funnel`, `sheltered-grasp`, `ramp-sort`, and `domino-single` transfer almost perfectly from understanding to planning (85–100%). GPT-5.5 maintains strong performance on geometry, dynamics, and causal-action categories, while stability reasoning (P7) remains inconsistent across models. Overall, the main challenge is action sequencing rather than recognizing the constraint itself (Appendix V).

B. Closed-Loop Manipulation Execution (Q3)

1) Reasoning-to-Action Execution (Stage 3): We extend the above agentic framework to translate reasoning into executable manipulation actions. In Stage 3, we use a closed-loop **ReAct (Reasoning + Acting)** framework where the VLM receives multi-view images, proprioception, and force-torque signals, then selects predefined action primitives with iterative feedback after each step. In addition to using the predefined primitives, the models are free to combine them into more complex behaviors or define new primitives when required. We evaluate GPT-5.5 in both vision-only and privileged object-centric settings across representative tasks from all categories. Due to the high cost of interactive evaluation (A complete Stage 3 evaluation across 16 tasks costs approximately $1800), we benchmark only GPT-5.5 as an agent, reported as `IMBAgent` and `IMBAgent-obj` in Table II. To provide broader insights, we additionally evaluate end-to-end visuomotor policies on IMBENCH. Further implementation details are provided in Appendix O.

Closed-loop execution exposes the largest reasoning-to-action gap. Across sixteen evaluated tasks, GPT-5.5 achieves non-zero success on only three (Table X). `domino-single` succeeds in all runs because corrective pushes tolerate imprecise contact. Privileged object poses improve `domino-select` and `mirror-pick-place`, increasing the mean success rate from 11.3% to 18.8%. However, all tasks requir-

TABLE IV
OOD GENERALIZATION (TASK-WISE). SUCCESS RATE UNDER IN-DISTRIBUTION (ID) AND OUT-OF-DISTRIBUTION (OOD) SETTINGS, WITH ABSOLUTE DROP Δ.

		$\pi_{0.5}$			**GR00T 1.5**			**DP**		
		ID	**OOD**	**Δ**	**ID**	**OOD**	**Δ**	**ID**	**OOD**	**Δ**
P1	sheltered-grasp	0.10	0.00	-0.10	0.00	0.00	0.00	0.30	0.00	-0.30
P2	pendulum-grasp	0.28	0.40	+0.12	0.08	0.04	-0.04	0.35	0.00	-0.35
P3	domino-single	0.73	0.52	-0.21	0.25	0.15	-0.10	0.70	0.60	-0.10
P4	tool-retreive	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
P5	occluder-push	0.15	0.08	-0.07	0.00	0.00	0.00	0.50	0.40	-0.10
P6	slip-recovery	0.22	0.08	-0.14	0.00	0.00	0.00	0.44	0.56	+0.12
P7	balance-medium	0.71	0.12	-0.59	0.02	0.00	-0.02	0.28	0.04	-0.24
M	keyboard-typing	0.13	0.00	-0.13	0.04	0.00	-0.04	0.32	0.04	-0.28

ing precise alignment, timing, tool use, hidden-state reasoning, or balancing score 0% in both settings. In some cases, privileged poses even hurt reasoning: on `mirror-pick-place`, the agent follows reported coordinates directly without applying the required mirror transformation. Per-task rollout analysis is provided in Appendix W.

2) End-to-end manipulation policies: We evaluate three end-to-end visuomotor policies, trained on the 200 human demonstration trajectories. **Diffusion Policy (DP)** is trained from scratch for each task, while $\pi_{0.5}$ and **GR00T N 1.5** (vision–language–action policies) are evaluated in zero-shot and finetuned settings. We evaluate two regimes: (1) **Zero-shot (ZS)**, using publicly available checkpoints without task-specific finetuning applied only to $\pi_{0.5}$ and GR00T N 1.5; and (2) **expert-demonstration training**, where all three approaches are trained/finetuned on curated IMBENCH data. For tasks where teleoperation is infeasible, data is collected using manually designed scripted policies. Training configurations are kept as close as possible to the original implementations. Full architecture and training details are in Appendix P.

Zero-shot VLA performance is extremely low (≤ 0.02 mean), and useful behavior only emerges after task-specific training (Table XV). Diffusion Policy performs best on mirror reasoning and contact-tolerant tasks, including `mirror-pick-place`, `domino-select`, `occluder-push`, and `slip-recovery`. Finetuned $\pi_{0.5}$ mainly succeeds on balancing tasks such as `balance-medium` and `balance-hard`. Several tasks remain unsolved by all policies, including `cup-inversion`, `shape-stack`, `stack-collapse-recovery`, `packing`, `mass-sort`, and both cracker-box grasp variants. Detailed task-level analysis is provided in Appendix X.

C. Out-of-distribution generalization

Changing the main physical factor of each task significantly reduces performance (Table IV) on tasks with the highest in-distribution scores. This suggests that policies rely heavily on memorized patterns rather than robust physical understanding. For example, `balance-medium` drops from 0.71 to 0.12 after a center-of-mass perturbation, while `keyboard-typing` falls from 0.32 to 0.04. Contact-tolerant tasks such as `domino-single` remain relatively robust. A few tasks show small improvements under perturbation, but overall performance degrades.

D. Discussion

Our results highlight three main findings. First, VLMs are much better at recognizing physical constraints than converting them into executable action plans. Tasks with simple geometric reasoning transfer well from understanding to planning, while tasks requiring ordered or timed actions often fail. Second, policy learning succeeds mainly on contact-tolerant tasks and a small number of structured behaviors such as mirror reasoning. Tasks requiring precise alignment, grasp mechanics, reorientation, or balancing remain unsolved across nearly all policies. Third, learned behaviors generalize poorly outside the training distribution. Even the strongest balancing policy drops sharply under small center-of-mass changes, and category-level performance remains low across geometry, dynamics, and stability tasks. Together, these results suggest that intuitive manipulation remains a missing capability axis that is not captured by current physical reasoning or scene understanding benchmarks.

V. LIMITATIONS

IMBench is currently limited to simulation and evaluates only Franka (widely-supported embodiment) with parallel-jaw and suction grippers, excluding dexterous hands, mobile platforms, and humanoids. Task descriptions are fixed, tasks are low-to-mid horizon,

and deformable object manipulation is not considered. While we release force-torque and tactile signals, the evaluated baselines do not use tactile or force-feedback modalities; principled integration of such signals remains an open direction that may improve performance. Finally, we focus on end-to-end vision-language-action baselines without privileged state information, emphasizing generalist agents and policies rather than task-specific engineered systems such as TAMP or long-horizon planners.

VI. Conclusion

We introduce IMBench, a benchmark of 35 manipulation tasks that test whether models can turn physical understanding into action. Tasks are organized around an Understand–Infer–Act loop and grouped by the intuition each one probes. We release ~14K curated demonstrations, a three-stage curation pipeline, and a scripted-demonstration generator, and we evaluate both high-level reasoning and low-level execution on current foundation models.

Three findings stand out. Frontier VLMs identify task constraints with moderate accuracy but propose valid plans less often. State-of-the-art VLAs score near zero zero-shot and perform poorly overall even after in-domain fine-tuning. And under OOD shifts along each task's target physical axis, all policies drop sharply. We position intuitive manipulation as a missing axis of capability, orthogonal to the dimensions current benchmarks reward, and release IMBench as a measurement instrument to help the field localize where intuitive physical intelligence lives in current systems.

ACKNOWLEDGMENTS

REFERENCES

[1] Kelsey R Allen, Kevin A Smith, and Joshua B Tenenbaum. Rapid trial-and-error learning with simulation supports flexible tool use and physical reasoning. *Proceedings of the National Academy of Sciences*, 117(47):29302–29310, 2020.

[2] Tayfun Ates, M Ateşoğlu, Çağatay Yiğit, Ilker Kesen, Mert Kobas, Erkut Erdem, Aykut Erdem, Tilbe Goksun, and Deniz Yuret. Craft: A benchmark for causal reasoning about forces and interactions. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2602–2627, 2022.

[3] Anton Bakhtin, Laurens van der Maaten, Justin Johnson, Laura Gustafson, and Ross Girshick. Phyre: A new benchmark for physical reasoning. *Advances in Neural Information Processing Systems*, 32, 2019.

[4] Hritik Bansal, Zongyu Lin, Tianyi Xie, Zeshun Zong, Michal Yarom, Yonatan Bitton, Chenfanfu Jiang, Yizhou Sun, Kai-Wei Chang, and Aditya Grover. Videophy: Evaluating physical commonsense for video generation. In *International Conference on Learning Representations*, volume 2025, pages 102075–102121, 2025.

[5] Fabien Baradel, Natalia Neverova, Julien Mille, Greg Mori, and Christian Wolf. Cophy: Counterfactual learning of physical dynamics. *arXiv preprint arXiv:1909.12000*, 2019.

[6] Daniel M Bear, Elias Wang, Damian Mrowca, Felix J Binder, Hsiao-Yu Fish Tung, RT Pramod, Cameron Holdaway, Sirui Tao, Kevin Smith, Fan-Yun Sun, et al. Physion: Evaluating physical prediction from vision in humans and machines. *arXiv preprint arXiv:2106.08261*, 2021.

[7] Shaofei Cai, Bowei Zhang, Zihao Wang, Xiaojian Ma, Anji Liu, and Yitao Liang. GROOT-1.5: Learning to follow multi-modal instructions from weak supervision. In *Multi-modal Foundation Model meets Embodied AI Workshop @ ICML2024*, 2024. URL https://openreview.net/forum?id=zxdi4Kdfjq.

[8] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14455–14465, 2024.

[9] Zhenfang Chen, Kexin Yi, Yunzhu Li, Mingyu Ding, Antonio Torralba, Joshua B Tenenbaum, and Chuang Gan. Comphy: Compositional physical reasoning of objects and events from videos. *arXiv preprint arXiv:2205.01089*, 2022.

[10] An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. Spatialrgpt: Grounded spatial reasoning in vision-language models. *Advances in Neural Information Processing Systems*, 37: 135062–135093, 2024.

[11] Anoop Cherian, Radu Corcodel, Siddarth Jain, and Diego Romeres. Llmphy: Complex physical reasoning using large language models and world models. *arXiv preprint arXiv:2411.08027*, 2024.

[12] Nikita Chernyadev, Nicholas Backshall, Xiao Ma, Yunfan Lu, Younggyo Seo, and Stephen James. Bigym: A demo-driven mobile bi-manual manipulation benchmark. In *Conference on Robot Learning*, pages 4201–4217. PMLR, 2025.

[13] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

[14] Wei Chow, Jiageng Mao, Boyi Li, Daniel Seita, Vitor Campagnolo Guizilini, and Yue Wang. Physbench: Benchmarking and enhancing vision-language models for physical world understanding. In *International Conference on Learning Representations*, volume 2025, pages 97959–98108, 2025.

[15] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[16] Mengfei Du, Binhao Wu, Zejun Li, Xuan-Jing Huang, and Zhongyu Wei. Embspatial-bench: Benchmarking spatial understanding for embodied tasks with large vision-language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 346–355, 2024.

[17] Jiafei Duan, Samson Yu, Soujanya Poria, Bihan Wen, and Cheston Tan. Pip: Physical interaction prediction via mental simulation with span selection. In *European Conference on Computer Vision*, pages 405–421. Springer, 2022.

[18] George Jiayuan Gao, Tianyu Li, Junyao Shi, Yihan Li, Zizhe Zhang, Nadia Figueroa, and Dinesh Jayaraman. Vlmgineer: Vision language models as robotic toolsmiths. *arXiv preprint arXiv:2507.12644*, 2025.

[19] Jensen Gao, Bidipta Sarkar, Fei Xia, Ted Xiao,

Jiajun Wu, Brian Ichter, Anirudha Majumdar, and Dorsa Sadigh. Physically grounded vision-language models for robotic manipulation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12462–12469. IEEE, 2024.

[20] Caelan Reed Garrett, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proceedings of the international conference on automated planning and scheduling*, volume 30, pages 440–448, 2020.

[21] Caelan Reed Garrett, Rohan Chitnis, Rachel Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems*, 4(1): 265–293, 2021.

[22] Rohit Girdhar and Deva Ramanan. Cater: A diagnostic dataset for compositional actions and temporal reasoning. *arXiv preprint arXiv:1910.04744*, 2019.

[23] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017.

[24] Thomas L Griffiths, Frederick Callaway, Michael B Chang, Erin Grant, Paul M Krueger, and Falk Lieder. Doing more with less: meta-reasoning and meta-learning in humans and machines. *Current Opinion in Behavioral Sciences*, 29:24–30, 2019.

[25] Minho Heo, Youngwoon Lee, Doohyun Lee, and Joseph J Lim. Furniturebench: Reproducible real-world benchmark for long-horizon complex manipulation. *The International Journal of Robotics Research*, 44(10-11):1863–1891, 2025.

[26] Suning Huang, Boyuan Chen, Huazhe Xu, and Vincent Sitzmann. Dittogym: Learning to control soft shape-shifting robots. *arXiv preprint arXiv:2401.13231*, 2024.

[27] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. In *Conference on Robot Learning*, pages 540–562. PMLR, 2023.

[28] Yixuan Huang, Bowen Li, Vaibhav Saxena, Yichao Liang, Utkarsh Aashu Mishra, Liang Ji, Lihan Zha, Jimmy Wu, Nishanth Kumar, Sebastian Scherer, et al. Kinder: A physical reasoning benchmark for robot learning and planning. *arXiv preprint arXiv:2604.25788*, 2026.

[29] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[30] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

[31] Baoxiong Jia, Yixin Chen, Huangyue Yu, Yan Wang, Xuesong Niu, Tengyu Liu, Qing Li, and Siyuan Huang. Sceneverse: Scaling 3d vision-language learning for grounded scene understanding. In *European Conference on Computer Vision*, pages 289–310. Springer, 2024.

[32] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: Robot manipulation with multimodal prompts. 2023.

[33] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017.

[34] Leslie P. Kaelbling and Tomas Lozano-Perez. Integrated task and motion planning in belief space, July 2013. ISSN 0278-3649. URL http://hdl.handle.net/1721.1/87038.

[35] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

[36] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[37] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

[38] James R. Kubricht, Keith J. Holyoak, and Hongjing Lu. Intuitive physics: Current research and controversies. *Trends in Cognitive Sciences*, 21(10):749–759, 2017. ISSN 1364-6613. doi: https://doi.org/10.1016/j.tics.2017.06.002. URL https://www.sciencedirect.com/science/article/pii/S1364661317301262.

[39] Fabien Lagriffoul, {Neil T.} Dantam, Caelan Garrett, Aliakbar Akbari, Siddharth Srivastava, and {Lydia E.} Kavraki. Platform-independent benchmarks for task and motion planning. *IEEE Robotics and Automation Letters*, 3(4):3765–3772, October 2018. ISSN 2377-3766. doi: 10.1109/LRA.2018.2856701.

[40] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40:e253, 2017.

[41] 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 *Conference on Robot Learning*, pages 80–93. PMLR, 2023.

[42] Shiqian Li, Kewen Wu, Chi Zhang, and Yixin Zhu. I-phyre: Interactive physical reasoning. In *International Conference on Learning Representations*, volume 2024, pages 28195–28215, 2024.

[43] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

[44] Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-world robotic manipulation through mark-based visual prompting. *arXiv preprint arXiv:2403.03174*, 2024.

[45] Ruiyuan Lyu, Jingli Lin, Tai Wang, Shuai Yang, Xiaohan Mao, Yilun Chen, Runsen Xu, Haifeng Huang, Chenming Zhu, Dahua Lin, et al. Mmscan: A multi-modal 3d scene dataset with hierarchical grounded language annotations. *Advances in Neural Information Processing Systems*, 37:50898–50924, 2024.

[46] Xiaojian Ma, Silong Yong, Zilong Zheng, Qing Li, Yitao Liang, Song-Chun Zhu, and Siyuan Huang. Sqa3d: Situated question answering in 3d scenes. *arXiv preprint arXiv:2210.07474*, 2022.

[47] Arjun Majumdar, Anurag Ajay, Xiaohan Zhang, Pranav Putta, Sriram Yenamandra, Mikael Henaff, Sneha Silwal, Paul Mcvay, Oleksandr Maksymets, Sergio Arnaud, Karmesh Yadav, Qiyang Li, Ben Newman, Mohit Sharma, Vincent Berges, Shiqi Zhang, Pulkit Agrawal, Yonatan Bisk, Dhruv Batra, Mrinal Kalakrishnan, Franziska Meier, Chris Paxton, Sasha Sax, and Aravind Rajeswaran. Openeqa: Embodied question answering in the era of foundation models. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[48] Michael Matthews, Michael Beukman, Chris Lu, and Jakob Foerster. Kinetix: Investigating the training of general agents through open-ended physics-based control tasks. In *International Conference on Learning Representations*, volume 2025, pages 58515–58564, 2025.

[49] Robert McCarthy, Daniel CH Tan, Dominik Schmidt, Fernando Acero, Nathan Herr, Yilun Du, Thomas G Thuruthel, and Zhibin Li. Towards generalist robot learning from internet video: A survey. *Journal of Artificial Intelligence Research*, 83, 2025.

[50] Michael McCloskey, Allyson Washburn, and Linda Felch. Intuitive physics: The straight-down belief and its origin. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 9:636–649, 10 1983. doi: 10.1037/0278-7393.9.4.636.

[51] 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*, 7(3):7327–7334, 2022.

[52] Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations, 2021. URL https://arxiv.org/abs/2107.14483.

[53] Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv preprint arXiv:2406.02523*, 2024.

[54] Soroush Nasiriany, Fei Xia, Wenhao Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny Driess, Ayzaan Wahid, Zhuo Xu, et al. Pivot: Iterative visual prompting elicits actionable knowledge for vlms. *arXiv preprint arXiv:2402.07872*, 2024.

[55] Dantong Niu, Yuvan Sharma, Giscard Biamby, Jerome Quenum, Yutong Bai, Baifeng Shi, Trevor Darrell, and Roei Herzig. Llarva: Vision-action instruction tuning enhances robot learning. In *Conference on Robot Learning*, pages 3333–3355. PMLR, 2025.

[56] Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International*

Conference on Learning Representations, volume 2025, pages 94937–94982, 2025.

[57] Maitreya Patel, Tejas Gokhale, Chitta Baral, and Yezhou Yang. Cripp-vqa: Counterfactual reasoning about implicit physical properties via video question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9856–9870, 2022.

[58] 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.

[59] Nazneen Fatema Rajani, Rui Zhang, Yi Chern Tan, Stephan Zheng, Jeremy Weiss, Aadit Vyas, Abhijit Gupta, Caiming Xiong, Richard Socher, and Dragomir Radev. Esprit: Explaining solutions to physical reasoning tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7906–7917, 2020.

[60] Ronan Riochet, Mario Ynocente Castro, Mathieu Bernard, Adam Lerer, Rob Fergus, Véronique Izard, and Emmanuel Dupoux. Intphys: A framework and benchmark for visual intuitive physics reasoning. *arXiv preprint arXiv:1803.07616*, 2018.

[61] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749, 2020.

[62] Arth Shukla, Stone Tao, and Hao Su. Maniskillhab: A benchmark for low-level manipulation in home rearrangement tasks. In *International Conference on Learning Representations*, volume 2025, pages 15288–15317, 2025.

[63] Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2998–3009, 2023.

[64] Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

[65] 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. *arXiv preprint arXiv:2405.12213*, 2024.

[66] Hsiao-Yu Tung, Mingyu Ding, Zhenfang Chen, Daniel Bear, Chuang Gan, Josh Tenenbaum, Dan Yamins, Judith Fan, and Kevin Smith. Physion++: Evaluating physical scene understanding that requires online inference of different physical properties. *Advances in Neural Information Processing Systems*, 36:67048–67068, 2023.

[67] Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.

[68] Andong Wang, Bo Wu, Sunli Chen, Zhenfang Chen, Haotian Guan, Wei-Ning Lee, Li Erran Li, and Chuang Gan. Sok-bench: A situated video reasoning benchmark with aligned open-world knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13384–13394, 2024.

[69] Chen Wang, Linxi Fan, Jiankai Sun, Ruohan Zhang, Li Fei-Fei, Danfei Xu, Yuke Zhu, and Anima Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. In *Conference on Robot Learning*, pages 201–221. PMLR, 2023.

[70] Shu Wang, Muzhi Han, Ziyuan Jiao, Zeyu Zhang, Ying Nian Wu, Song-Chun Zhu, and Hangxin Liu. Llm^ 3: Large language model-based task and motion planning with motion failure reasoning. In *2024 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 12086–12092. IEEE, 2024.

[71] Xingrui Wang, Wufei Ma, Angtian Wang, Shuo Chen, Adam Kortylewski, and Alan Yuille. Compositional 4d dynamic scenes understanding with physics priors for video question answering. In *International Conference on Learning Representations*, volume 2025, pages 93688–93700, 2025.

[72] Yi Wang, Jiafei Duan, Dieter Fox, and Siddhartha Srinivasa. Newton: Are large language models capable of physical reasoning? In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9743–9758, 2023.

[73] Bo Wu, Shoubin Yu, Zhenfang Chen, Joshua B Tenenbaum, and Chuang Gan. Star: A benchmark for situated reasoning in real-world videos. *arXiv preprint arXiv:2405.09711*, 2024.

[74] Jiajun Wu, Joseph J Lim, Hongyi Zhang, Joshua B

Tenenbaum, and William T Freeman. Physics 101: Learning physical object properties from unlabeled videos. In *BMVC*, volume 2, page 7, 2016.

[75] Qiucheng Wu, Handong Zhao, Michael Saxon, Trung Bui, William Yang Wang, Yang Zhang, and Shiyu Chang. Vsp: Assessing the dual challenges of perception and reasoning in spatial planning tasks for vlms. *arXiv preprint arXiv:2407.01863*, 2024.

[76] Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10632–10643, 2025.

[77] Rui Yang, Hanyang Chen, Junyu Zhang, Mark Zhao, Cheng Qian, Kangrui Wang, Qineng Wang, Teja Venkat Koripella, Marziyeh Movahedi, Manling Li, et al. Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. In *International Conference on Machine Learning*, pages 70576–70631. PMLR, 2025.

[78] Angen Ye, Zeyu Zhang, Boyuan Wang, Xiaofeng Wang, Dapeng Zhang, and Zheng Zhu. Vla-r1: Enhancing reasoning in vision-language-action models. *arXiv preprint arXiv:2510.01623*, 2025.

[79] Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B Tenenbaum. Clevrer: Collision events for video representation and reasoning. *arXiv preprint arXiv:1910.01442*, 2019.

[80] Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. In *Conference on Robot Learning*, pages 3157–3181. PMLR, 2025.

[81] Shiduo Zhang, Zhe Xu, Peiju Liu, Xiaopeng Yu, Yuan Li, Qinghui Gao, Zhaoye Fei, Zhangyue Yin, Zuxuan Wu, Yu-Gang Jiang, et al. Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11142–11152, 2025.

[82] Zhigen Zhao, Shuo Cheng, Yan Ding, Ziyi Zhou, Shiqi Zhang, Danfei Xu, and Ye Zhao. A survey of optimization-based task and motion planning: From classical to learning approaches. *IEEE/ASME Transactions On Mechatronics*, 30 (4):2799–2825, 2024.

[83] Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3d-vla: A 3d vision-language-action generative world model. *arXiv preprint arXiv:2403.09631*, 2024.

[84] Yuke Zhu, Josiah Wong, Ajay Mandlekar, and Roberto Martín-Martín. robosuite: A modular simulation framework and benchmark for robot learning. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10950–10957. IEEE, 2020. doi: 10.1109/IROS45743.2020.9340558.

[85] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.

APPENDIX

IMBENCH differs from prior benchmarks by explicitly evaluating the full pipeline from physical reasoning to executable manipulation. It separately measures VLM constraint understanding, plan generation, and closed-loop execution. In addition, it introduces bimanual coordination tasks with expert demonstrations, enabling evaluation of both reasoning and control. We additionally release tactile and force/torque data to support multimodal manipulation research and improve performance on IMBENCH.

This appendix gives the full specification of every IMBENCH task. Each entry lists: phenomenon category, embodiment, end-effector configuration, the physical axis the task isolates, the success predicate, the reset-time randomization ranges, the canonical sub-goal sequence used by the Stage-2 oracle, the demonstration counts after curation, and the mean episode length. We release a subset of the dataset (10 episodes) via Hugging Face for review purposes. To facilitate easy inspection, we provide utilities that allow reviewers to directly visualize the dataset through a web link without requiring any installation. The complete benchmark and full dataset will be released upon acceptance. Each task description below includes an associated link that redirects to the LeRobot visualizer for interactive inspection.

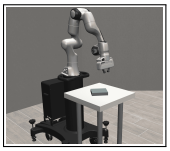
A. Master catalog

For each task we target 200 curated teleoperation and 200 curated scripted trajectories. For the five tasks where teleoperation is infeasible (T10–T14), 400 scripted trajectories per task are collected to compensate. After the three-stage curation pipeline, the released dataset contains 6,000 teleoperation + 8,000 scripted = 14,000 trajectories ($\approx$ 14K total).

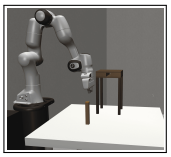
B. Per-category specifications

For each task we report: **Setup** (initial scene), **Goal** (success condition), **Intuition bottleneck** (the physical fact a model must infer), and **Canonical plan** (the Stage-2 reference sub-goal sequence). Only quantities that change how the task is reasoned about are listed; spawn jitter, control gains, and episode budgets are deferred to the task configs. A representative initial-state image of each scene appears to the left of its description.

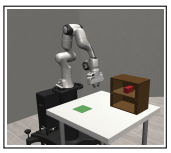
1) P1 — Geometry-constrained grasping:



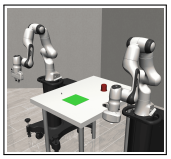
T01 `edge-slide`. *Setup:* a thin rectangular plate, lies flush on the table. *Goal:* lift the plate above the table. *Intuition:* a top-down jaw closure cannot close around this small thickness; the plate must first be slid to overhang the table edge to expose a graspable lip. *Canonical plan:* approach the plate, push it sideways until it overhangs the table edge, pinch the exposed lip, and lift. T01 edge-slide



T02 `recover-peg-insert`. *Setup:* a rod stands vertically; a through-slot present horizontally on a side board. *Goal:* drive the peg through $\approx 95\%$ of its length into the slot, aligned with the slot axis. *Intuition:* the slot is horizontal, the arm must grasp and align the peg to the axis of the slot. A one-time $\pm 10^\circ$ yaw perturbation, applied once the peg aligns with the slot. This forces the agent to think of correct geometric alignment and constrained insertion. *Canonical plan:* tip the rod onto its side against a nearby surface, regrasp it laterally so its long axis lies along the slot axis, then push it through. Realign if the line of approach is not correct. T02 recover-peg-insert



T03 `sheltered-grasp`. *Setup:* a cube sits in either the top or bottom story of a covered two-tier rack; the side face is open but the top is occluded. A goal tray sits on the table. The observations include camera views where the object is initially not visible, requiring the agent to first explore and then rely solely on the gripper camera to locate and pick the object. *Goal:* drop the cube in the tray. *Intuition:* only lateral approach through the open face is feasible; the gripper must yaw to enter the rack. *Canonical plan:* Explore the scene to locate object, yaw the gripper to face the rack's open side, enter laterally, grasp the cube, withdraw, and drop it in the tray. T03 sheltered-grasp

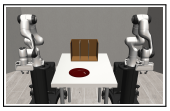


T04 `cup-inversion`. *Setup:* a handle-less cup starts upside-down on the table. Grippers are heterogeneous: one parallel-jaw, one suction. *Goal:* place the cup upright inside a goal region. *Intuition:* suction couples to the closed bottom but cannot achieve upright placement alone; the jaw cannot grasp the smooth rim. A reorienting suction-to-jaw handover is required. Toppling is not a reliable option since the cup just slides out. *Canonical plan:* attach suction to the cup's exposed (upward-facing) base, lift, rotate the cup so the opening faces up, hand it off to the jaw which grabs it by the rim, and place it inside the goal region. T04 cup-inversion

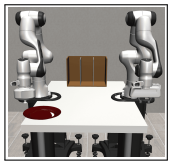
TABLE V

MASTER CATALOG OF IMBENCH TASKS. "DEMOS" LISTS CURATED COUNTS AFTER THE THREE-STAGE FILTERING PIPELINE. A DASH (–) IN THE TELEOP COLUMN DENOTES A TASK FOR WHICH TELEOPERATION IS INFEASIBLE WITH THE SPACEMOUSE INTERFACE.

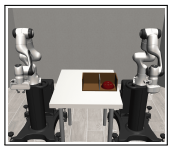
ID	Task	Cat.	Embodiment / EE	Demos (teleop / scripted)	Mean ep. (s)
T01	edge-slide	P1	1-arm / jaw	200 / 200	12.4
T02	recover-peg-insert	P1	1-arm / jaw	200 / 200	14.1
T03	sheltered-grasp	P1	1-arm / jaw	200 / 200	9.8
T04	cup-inversion	P1	het. bi / suc+jaw	200 / 200	18.6
T05	plate-shelf-hobi	P1	hom. bi / jaw+jaw	200 / 200	21.3
T06	plate-shelf-hetbi	P1	het. bi / suc+jaw	200 / 200	19.7
T07	bowl-table	P1	het. bi / suc+jaw	200 / 200	17.2
T08	cracker-box-suction	P1	1-arm / suction	200 / 200	11.7
T09	cracker-box-jaw	P1	1-arm / jaw	200 / 200	13.2
T10	cube-toss	P2	1-arm / jaw	– / 400	4.6
T11	pendulum-grasp	P2	1-arm / jaw	– / 400	7.3
T12	linear-intercept	P2	1-arm / jaw	– / 400	5.1
T13	bounce-intercept	P2	1-arm / jaw	– / 400	6.8
T14	galileo-ramp-drop	P2	1-arm / jaw	– / 400	8.4
T15	slide-catch	P2	1-arm / jaw	200 / 200	9.6
T16	ramp-sort	P2	1-arm / jaw	200 / 200	16.2
T17	domino-single	P3	1-arm / jaw	200 / 200	6.3
T18	domino-select	P3	1-arm / jaw	200 / 200	8.1
T19	seesaw-balance	P3	1-arm / jaw	200 / 200	11.5
T20	indirect-push	P3	1-arm / jaw	200 / 200	9.4
T21	gap-funnel	P3	1-arm / jaw	200 / 200	10.7
T22	ball-cage-escape	P3	1-arm / jaw	200 / 200	13.9
T23	tool-retrieve	P4	1-arm / jaw	200 / 200	17.8
T24	cup-extract	P4	1-arm / jaw	200 / 200	15.4
T25	mass-sort	P5	1-arm / jaw	200 / 200	24.6
T26	occluder-push	P5	1-arm / jaw	200 / 200	14.3
T27	slip-recovery	P6	1-arm / jaw	200 / 200	11.2
T28	stack-collapse-recovery	P6	1-arm / jaw	200 / 200	19.5
T29	pick-place-ycb	P6	1-arm / jaw	200 / 200	13.7
T30	shape-stack	P7	1-arm / jaw	200 / 200	16.8
T31	balance-medium	P7	1-arm / jaw	200 / 200	18.9
T32	balance-hard	P7	1-arm / jaw	200 / 200	21.6
T33	packing	P7	1-arm / jaw	200 / 200	25.4
M01	mirror-pick-place	M	1-arm / jaw	200 / 200	8.7
M02	keyboard-typing	M	1-arm / jaw	200 / 200	28.3



T05 `plate-shelf-hobi`. *Setup:* A cutlery plate is positioned next to a center-shelf slot. Both arms carry parallel-jaw grippers. *Goal:* Insert the plate on edge into the shelf slot. *Intuition:* The flat plate cannot be grasped by a single arm. A symmetric two-handed pick is also infeasible, since mid air transfer is not possible without grasping, which renders final shelf placement impossible. The plate's slight incline geometry enables an alternative approach: one robotic arm presses the plate near its circumference to tilt it, creating a graspable entry for the other arm. *Canonical Plan:* Press the plate with one arm around the corner to create an edge. Grasp the plate with the other arm and transport it to the center-shelf for insertion. T05 plate-shelf-hobi

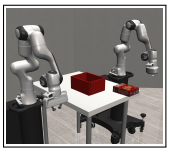


T06 `plate-shelf-hetbi`. *Setup:* Similar to T05, but the two arms carry different end-effectors: one suction cup, one parallel-jaw gripper. *Goal:* Insert the plate on edge into the shelf slot. *Intuition:* The asymmetric grippers enforce a role split. The suction cup lifts the plate by its broad face, replacing the press action needed in T05. The parallel-jaw gripper then grasps the plate edge. An explicit suction-to-jaw handover is required, as the suction cup cannot reach into the shelf to perform final insertion. *Canonical Plan:* Suction arm lifts the plate by its broad face. Jaw arm grasps the plate edge and inserts it into the shelf slot. T06 plate-shelf-hetbi

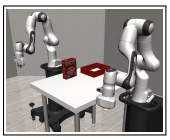


T07 `bowl-table`. *Setup:* a bowl starts upside-down in a rack. Grippers: parallel-jaw + suction.

Goal: place the bowl upright on the table. *Intuition:* The jaw cannot grasp the bowl since no feasible affordances are available. The suction can extract the bowl from the rack but cannot reorient it cleanly on the smooth exterior; jaw must take over on the bowl rim to invert and lower it to place it in goal position. *Canonical plan:* use the suction to lift the bowl out of the rack, attach jaw to its (now exposed) rim, rotate it to upright, and set it down on the table. T07 bowl-table

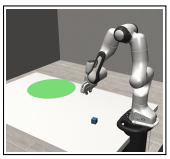


T08 `cracker-box-suction`. *Setup:* A cracker box stands upright on the table. A horizontal slot fixture is within reach. Only the suction arm is active; the parallel-jaw arm is frozen. *Goal:* Insert the box into the slot with its centroid aligned to the slot interior. *Intuition:* Grasping the narrow side is inefficient and requires extensive readjustment. Suction couples reliably to the broad cardboard face, making it the preferred approach. *Canonical Plan:* Align the suction cup to the broad side of the box, transport and realign to the slot and descend uniformly into it. T08 cracker-box-suction

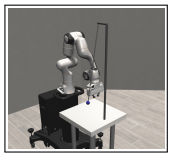


T09 `cracker-box-jaw`. *Setup:* Same as T08, but only the parallel-jaw arm is active; the suction arm is frozen. *Goal:* Insert the box into the slot. *Intuition:* The jaw can apply lateral forces but cannot directly fit the box into the tight slot outline. The box must be partially inserted first, then adjusted through knocking, shifting, and aligning actions to achieve final alignment. *Canonical Plan:* Grasp the box by its narrow side (flat-side grasping is not feasible with the jaw). Partially insert the box into the slot and then adjust laterally to achieve full insertion and alignment. T09 cracker-box-jaw

2) P2 — Dynamics and trajectory prediction:

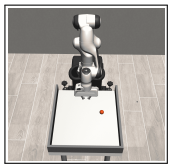


T10 `cube-toss`. *Setup:* A gripper holds a cube; the goal zone lies beyond the arm's reach. *Goal:* The cube comes to rest inside the goal zone after release. *Intuition:* Since the goal is out of reach, the cube must be thrown with sufficient velocity to reach the zone. This requires understanding projectile motion under gravity to determine appropriate throw velocity and direction. *Canonical Plan:* Lift the cube and throw it toward the goal zone with sufficient velocity and direction to land inside. T10 cube-toss

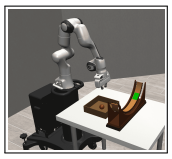


T11 `pendulum-grasp`. *Setup:* A sphere swings on an undamped pendulum at moderate amplitude. *Goal:* Grasp the bob at the swing apex. *Intuition:* Angular velocity reaches zero only at the swing extremes. Grasp timing must align with the apex phase when the bob's velocity is zero. This requires understanding periodic motion and predicting when the bob reaches the turning point. *Canonical Plan:* Track the pendulum motion and time the grasp to coincide with the swing apex, where the bob momentarily stops before reversing direction. T11 pendulum-grasp

T12 `linear-intercept`. *Setup:* A ball rolls across the table at constant velocity, starting near one edge and heading toward the table center. *Goal:* Grasp the ball and hold it above the table for 30 consecutive steps. *Intuition:* Linear extrapolation of the constant-velocity trajectory determines the future position and timing of the ball. The gripper must be positioned at the interception point to grasp the ball when it arrives. *Canonical Plan:* Predict the ball's trajectory using its current velocity. Position the gripper at the future interception point and close the gripper as the ball arrives. Lift and hold the grasped ball above the table. T12 linear-intercept

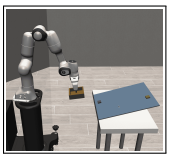


T13 `bounce-intercept`. *Setup:* A ball is launched at a U-shaped wall enclosure and ricochets at least once before reaching the catchable zone. *Goal:* Grasp the ball and hold it above the table for 30 consecutive steps. *Intuition:* Predicting the post-bounce trajectory requires understanding how the ball's pre-bounce heading and the wall geometry jointly determine the final direction. *Canonical Plan:* Predict the ball's trajectory after bouncing off the wall using its initial heading and the wall normals. Position the gripper at the predicted interception point. Close the gripper as the ball arrives and lift it. T13 bounce-intercept

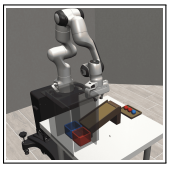


T14 `galileo-ramp-drop`. *Setup:* A steel ball sits in an open container next to a frictionless symmetric ramp. *Goal:* Release the ball on the ramp so it arrives at a goal patch on the far side at near-zero velocity. *Intuition:* Energy conservation on the frictionless ramp determines the release height needed to achieve

near-zero terminal velocity at the goal patch. The ball's potential energy at release must equal its kinetic energy at the patch to minimize arrival speed. *Canonical Plan:* Approximate the required release height using energy conservation and the ramp geometry. Bring the ball at that height and release it. T14 galileo-ramp-drop

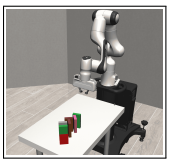


T15 `slide-catch`. *Setup:* An incline tilts from $10°$ to $55°$. Cubes with different friction coefficients sit on the incline. A graspable bin is on the table. *Goal:* Catch all cubes in the bin. *Intuition:* Each cube slides when the incline angle exceeds its friction limit, causing them to slide off at different times. The cubes closer to the bottom slide first. The bin must be positioned at the hinge edge at the right time to catch each cube in sequence. *Canonical Plan:* Position the bin to catch the first (lowest) cube as it slides off. After catching, reposition the bin for the next cube, repeating until all are captured. T15 slide-catch

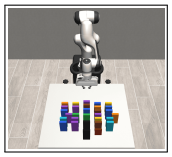


T16 `ramp-sort`. *Setup:* Multiple colored balls spawn in a low-walled box. A tilted ramp routes them downward into two bins. Barriers for goal are invisible to balls but solid to the arm. *Goal:* Every ball lands in its correct bin based on color. *Intuition:* Color-dependent rolling trajectories separate red balls from blue balls down the ramp. Any misdirected placement is irreversible. Release pose and timing determine the ball's trajectory down the ramp, so they must be chosen carefully to reach the correct bin. *Canonical Plan:* For each ball, determine the correct target bin based on its color. Release the ball at a pose such that it rolls down the ramp and lands in the correct bin. T16 ramp-sort

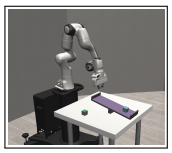
3) P3 — Causal and indirect action:



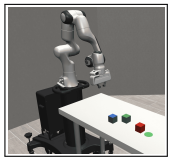
T17 `domino-single`. *Setup:* a chain of dominoes leads to a red target slab. *Goal:* the cascade displaces the target by at least 5 cm, *and the gripper never touches the target*. *Intuition:* cascade physics — toppling the first domino must propagate through the chain. Touching the target directly is an immediate, latched failure. *Canonical plan:* approach the first domino in the chain, push it toward its neighbour, and wait for the cascade to reach the target. T17 domino-single



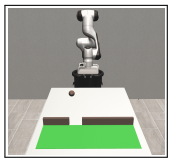
T18 `domino-select`. *Setup:* Five domino chains are present. Only one cascades fully to the target. Others contain failures: gap (domino offset perpendicular), drift (spacing increases along the chain), or the chain points to the wrong target. *Goal:* Displace the target domino via chain reaction. *Intuition:* The correct chain must be identified by visual inspection before triggering. Determine whether the chosen cascade will successfully reach the target. *Canonical Plan:* Select the structurally intact chain. Knock down the first domino in that chain. T18 domino-select



T19 `seesaw-balance`. *Setup:* A block sits at a random offset on one arm of a pivoted seesaw. A counterweight with unknown mass starts off the seesaw. Both masses are randomized. *Goal:* Hold the seesaw tilted within $5°$ of level. *Intuition:* Torque balance requires $m_{\text{target}} \cdot r_{\text{target}} = m_{\text{counter}} \cdot r_{\text{counter}}$. Since masses are unknown, the counterweight position must be found through trial and adjustment by observing the seesaw's tilt. *Canonical Plan:* Place the counterweight at an estimated offset. Observe the seesaw's tilt and adjust the counterweight position accordingly. Iterate until balance is achieved. T19 seesaw-balance

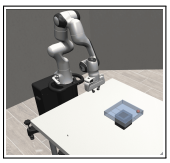


T20 `indirect-push`. *Setup:* A scatter of colored blocks and a red target block sit on the table. The goal zone is beyond the arm's direct reach. *Goal:* Drive the target into the goal zone. *Intuition:* Direct contact with the target is not possible from the arm's reach. The scattered blocks must be arranged into a contiguous chain to transmit force. Pushing the chain propagates the impulse through to the target. *Canonical Plan:* Arrange the colored blocks into a contiguous chain connecting the target to the goal zone. Push the first block in the chain so the impulse propagates through to the target, driving it into the goal zone. T20 indirect-push



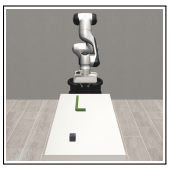
T21 `gap-funnel`. *Setup:* A static wall with a single narrow gap runs across the table. A medium sphere is positioned in front of the wall. *Goal:* Push the sphere through the gap into the goal zone (outside robot's workspace). *Intuition:* The sphere must first be steered to align with the gap, then pushed through with appropriate velocity and direction. Path planning must account for the barrier and the narrow passage. *Canonical Plan:* Align the sphere

with the gap by pushing it laterally into position. Once aligned, push the sphere through the gap with controlled velocity to reach the goal zone. T21 gap-funnel



T22 `ball-cage-escape`. *Setup:* A thin rectangular cage containing a ball sits on a narrow stool. The cage's bottom has a hole that is blocked by the stool. *Goal:* Extract the ball from the cage onto the table. *Intuition:* Lifting and reorienting the cage will allow the ball to escape through the hole. The reorientation must account for the ball's position inside the cage to ensure it falls out. *Canonical Plan:* Lift the cage off the stool. Tilt and reorient it to align the hole with the ball's position. Continue until the ball falls out onto the table. T22 ball-cage-escape

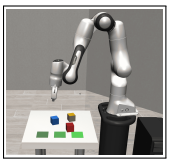
4) P4 — Tool use and augmented reach:



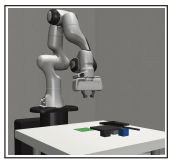
T23 `tool-retrieve`. *Setup:* an L-shaped hook is reachable; a cube target sits beyond the arm's normal reach. *Goal:* lift the cube above the table. *Intuition:* the tool extends reach, but the gripper cannot hold both tool and target : the tool must be released *before* the final grasp; releasing it after grasping the target is kinematically infeasible. *Canonical plan:* pick up the hook, drag the cube inward until it sits within normal reach, set the hook down, then grasp the cube and lift. T23 tool-retrieve

T24 `cup-extract`. *Setup:* A small cuboid is trapped inside an open-top cup. The handle points toward the robot. A goal region sits away from the cup. *Goal:* Extract the cuboid from the cup and place it in the goal region. *Intuition:* The cuboid cannot be grasped directly. Only the cup handle affords a clean grasp. The cup must be manipulated by its handle to dump the cuboid out. *Canonical Plan:* Grasp the cup by its handle. Tip and invert the cup to pour the cuboid out onto the table. Move the cuboid into the goal region. T24 cup-extract

5) P5 — Hidden state and active discovery:

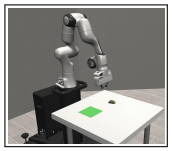


T25 `mass-sort`. *Setup:* Three visually identical cubes have hidden masses of 50 g, 100 g, and 200 g in randomized order. Three labeled zones on the table accept the cubes in light-to-heavy order from left to right. *Goal:* Place each cube in its correct zone according to mass. *Intuition:* Mass cannot be determined visually. It is observable only through wrist force and torque during a probing lift. Each cube must be probed before placement. Any wrong-zone placement is an immediate failure. *Canonical Plan:* Lift and probe each cube to estimate its mass. Sort the cubes by inferred mass. Place them in the zones in light-to-heavy order from left to right. T25 mass-sort

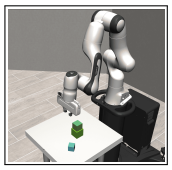


T26 `occluder-push`. *Setup:* One red target cube and two distractor cubes are each covered by a lightweight lid. A green goal patch sits elsewhere on the table. *Goal:* Place the red target cube on the goal patch. *Intuition:* The red cube cannot be identified until its lid is lifted or pushed aside. The agent must uncover each cube to locate the target. *Canonical Plan:* Lift or push aside the lids to reveal each cube. Identify the red target. Grasp the red cube and place it on the goal patch. T26 occluder-push

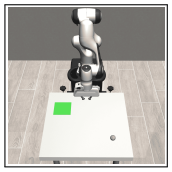
6) P6 — Reactive replanning:



T27 `slip-recovery`. *Setup:* A YCB object is placed on the table. The agent must pick and place it. Once during transport, the gripper unexpectedly opens for a brief window creating a synthetic failure. Sliding the object is not allowed. This tests state awareness and recovery behavior. *Goal:* Place the object inside the goal region. *Intuition:* The agent must detect the slip, re-grasp the object, and continue transport. *Canonical Plan:* Lift the object and transport it toward the goal. When the gripper opens, detect the slip. Re-grasp the object and resume task. T27 slip-recovery



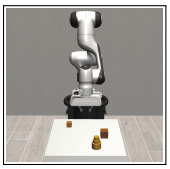
T28 `stack-collapse-recovery`. *Setup:* Three cubes of different sizes start pre-stacked. *Goal:* Rebuild a stable stack. *Intuition:* An external impulse knocks over the stack, dislodging either the top cube alone or both upper cubes in different directions. The agent must rebuild a stable stack, respecting the cube sizes and mass distribution to prevent tipping. *Canonical Plan:* Detect the collapse and gather the fallen cubes. Rebuild the stack ensuring stability. T28 stack-collapse-recovery



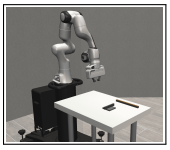
T29 `pick-place-ycb`. *Setup:* A single YCB object and a green goal patch sit on the table. *Goal:* Place the object at rest in the goal patch (after lifting it at least once). *Intuition:* When the lifted

object approaches the patch, the patch teleports to a new location on the table. After 1–3 relocations, the patch locks in place. The agent must replan and approach the locked patch. *Canonical Plan:* Lift the object and move it toward the goal patch. When the patch relocates, track its new position, replans the path, and approach again. Repeat until the patch locks, then complete the placement. T29 pick-place-ycb

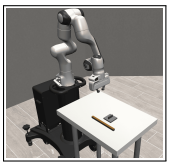
7) P7 — Stability and equilibrium:



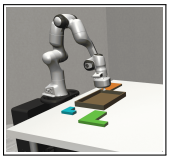
T30 `shape-stack`. *Setup:* A random subset of 3–5 mixed-shape objects on the table: cylinders, cubes, prisms, hex prisms, and cone frustums. *Goal:* Stack all objects so exactly one touches the table. *Intuition:* Stable stacking requires each upper piece's center of mass to lie within the support polygon of the base. Shape-dependent geometry determines which orientations are stable. *Canonical Plan:* Select the largest or most stable object as the base. Stack objects in order, ensuring each piece's center of mass aligns over the supporting base. T30 shape-stack



T31 `balance-medium`. *Setup:* A rod must be balanced on a narrow ridge of width 12 mm. *Goal:* The rod stands within 10° of vertical and remains in contact with the ridge. *Intuition:* The 12 mm ridge provides modest tolerance for initial tilt. The rod must be released with low angular velocity and its center of mass positioned above the ridge for stable balance. *Canonical Plan:* Position the rod on the ridge with minimal tilt and angular velocity. Release the rod and allow it to balance on its own. T31 balance-medium

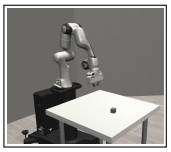


T32 `balance-hard`. *Setup:* A rod must be balanced on a narrow ridge of width 8 mm. *Goal:* The rod stands within 10° of vertical and remains in contact with the ridge. *Intuition:* The 8 mm ridge provides minimal tolerance. The rod must be released with very low residual angular velocity and its center of mass nearly perfectly positioned above the ridge for stable balance. *Canonical Plan:* Carefully position the rod on the ridge with minimal tilt and angular velocity. Release the rod and allow it to balance on its own. T32 balance-hard

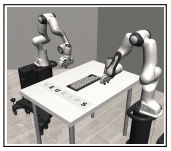


T33 `packing`. *Setup:* A rectangular bin and a set of L-shaped pieces obtained from guillotine cuts of a rectangle. A suction gripper is used. *Goal:* Pack every piece flat inside the bin with zero gaps. *Intuition:* The L-pieces originate from a guillotine partition of the bin's footprint, so a perfect zero-gap packing exists. The agent must recover the partition structure and orient each piece to fit its slot exactly. *Canonical Plan:* Determine how the L-pieces fit together to reconstruct the original rectangular partition. Orient and place each piece into its corresponding slot in the bin, working toward a complete zero-gap packing. T33 packing

8) Miscellaneous:



M01 `mirror-pick-place`. *Setup:* A single cube sits on the table. Every camera frame is rotated 180° each step. *Goal:* Lift the cube to a small height above the table. *Intuition:* The agent must distinguish between the observation frame (rotated) and the action frame (fixed). Direct imitation would invert the action direction. *Canonical Plan:* Track the cube's position in the rotated frame. Execute lift actions in the correct action frame, accounting for the 180° rotation. M01 mirror



M02 `keyboard-typing`. *Setup:* Two robotic arms face a QWERTY virtual keyboard. Arm 0 is scripted to type a subset of letters. Arm 1 is the agent. The letter partition between arms is randomized each episode. *Goal:* Type the target word in correct sequence, with each letter pressed and briefly held. *Intuition:* The agent must track which letters remain untyped and which it owns. Visual feedback on the keyboard indicates typed (bold) versus untyped (faint) letters. *Canonical Plan:* Observe which letters are marked as untyped. Identify your assigned letters from the remaining set. Press and hold each of your letters in the order required by the target word. M02 keyboard-typing

C. Teleoperation hardware and protocol

a) Single-arm.: Xbox-compatible gamepad, mapped to 6-DoF end-effector delta commands in the robot base frame. Translation magnitude 0.3 per tick, rotation magnitude 0.2 per tick. Gripper value is ramped via analog trigger; a snap-toggle to fully open/close is available on the A button with a boolean latch. **Bimanual tasks:** Same controller in arm-switching configuration: A pilot study with parallel two-handed control yielded poorer trajectory quality.

b) Filtering.: Actions are smoothed via an Exponential Moving Average (EMA) filter with α = 0.8, applied independently to each of the 6 motion channels (dx, dy, dz, droll, dpitch, dyaw) per arm. A per-step delta clamp of ±0.15 is applied after smoothing to limit abrupt transitions. Gripper channels pass through unfiltered to preserve snap-toggle responsiveness. Filter parameters are fixed across operators.

c) VR comparison.: We conducted a pilot study using META QUEST-based teleoperation on six representative tasks across 120 episodes. The mean post-filter success rate was 11.2%, compared to 58.7% achieved using an XBox controller on the same tasks. We attribute this difference to the smoother and more stable motion control provided by the XBox controller, particularly in the motion regimes dominant in IMBENCH. Based on these initial results, we used the XBox controller for full data collection, while retaining support for VR teleoperation within the benchmark.

D. Operator training, demographics, and consent

Operators completed 50-60 training trajectories on each of the tasks before contributing. Calibration is repeated at the start of each session. We collected from N=10 operators (10 male; ages 21-34; all university-affiliated graduate students or research staff with backgrounds in STEM).

E. Scripted oracle authoring

Each task has a hand-written policy. Oracles consume privileged simulator state and produce 6-DoF end-effector delta pose at, using the same action interface a deployed policy uses. Oracle actions are passed through the same filtering as teleop. Oracles are not motion planners; they encode task-specific strategies. For example, the `edge-slide` oracle:

```
def step(state):
    if not plate_at_edge(state):
        return slide_action(
            plate_pos,
            nearest_edge)
    elif not gripper_under_overhang(
            state):
        return descend_action(
            plate_overhang_pose)
    else:
        return close_and_lift_action()
```

F. Data Filtration

1) Human Annotation Dashboard: Prior to automated analysis, each recorded robotic episode is reviewed through a human annotation dashboard. A human annotator inspects the episode(comprising RGB-D frames, joint states, end-effector poses, and object positions and labels scene-level metadata such as task completion intent, object identities, and any immediately apparent recording artifacts) and assigns a quality rating on a scale of 1 to 5. Only episodes receiving a rating greater than or equal to 4 are passed to the automated filtration stage; all others are discarded immediately. This initial screening ensures that only demonstrations of sufficient quality enter the downstream pipeline.

2) VLM-Based Automated Filtering: Following human annotation, each episode is passed through an autonomous VLM-guided filtration agent built on a LangChain/LangGraph tool-calling framework. The agent executes a deterministic preflight sequence over every episode, invoking tools for metadata and field integrity validation, full-episode sensor signal reading, rule-based anomaly evaluation, signal-processing metric computation, and FFmpeg-based visual glitch detection via blur (Sobel gradient < 0.3) and scene-change (MAD score > 10.0) filters. Following this fixed pass, the agent enters a dynamic reasoning loop, issuing targeted follow-up tool calls such as high-FPS local frame inspection and per-channel signal plots until a verification gate confirms that completion status, anomaly evidence, sensor coverage, and signal-processing analyses are all satisfied. The agent produces a structured `EpisodeAnalysisReport` for each episode, flagging issues such as state discontinuities, action–response inconsistencies, index–timestamp mismatches, and visual anomalies, each with an associated severity and human-readable explanation.

3) Human Verification Dashboard: The agent-generated reports are surfaced through a second human verification dashboard, where annotators review the flagged episodes and their associated evidence including anomaly timestamps, sensor signal plots, and high-FPS frame windows before issuing a final accept or reject decision. This stage serves as a quality gate over the automated flagging, allowing annotators to override false positives and confirm true failures. Only episodes that clear this final human review are retained for training. Full prompt templates, tool schemas, verification gate logic, and filtration thresholds are provided in the supplementary material.

G. Modality schemas

All streams are aligned to simulator wall-clock at ingestion; camera intrinsics and extrinsics are stored under `meta/cameras`.

H. Custom MJCF assets

The following assets ship under IMBench:

- **Galileo ramp** (parametric angle $10^\circ - 60^\circ$, $\mu \in [0.10, 0.50]$).
- **Domino strips** (configurable inter-tile gap $0.6 \times$ –$1.4\times$ tile width; tile mass $0.02 - 0.08$ kg).

TABLE VI
PER-TRAJECTORY STREAMS WRITTEN BY THE IMBENCH LEROBOT RECORDER. RATES ARE CONFIGURABLE PER CATEGORY VIA `RECORDING.SAVE_FREQ.*`; VALUES SHOWN ARE TYPICAL DEFAULTS FOR A 100 HZ CONTROL LOOP. CAMERAS ARE STORED AS MP4 WITH PER-ROW `FRAME_INDEX.*` POINTERS IN PARQUET.

Stream	Shape per frame	Rate	Notes
Cameras (RGB, MP4-encoded)			
`rgb_exo_right_cam`	$(480, 640, 3)$ uint8	25 Hz	3rd-person right (`agentview`)
`rgb_exo_left_cam`	$(480, 640, 3)$ uint8	25 Hz	3rd-person left (`sideview`)
`rgb_ego_cam`	$(480, 640, 3)$ uint8	25 Hz	robot-view (`robot0_robotview`)
`rgb_gripper_cam`	$(480, 640, 3)$ uint8	25 Hz	wrist-mounted (`robot0_eye_in_hand`)
Proprioception			
`robot0_eef_pos`	$(3,)$ float32	100 Hz	EEF position, world frame, m
`robot0_eef_quat`	$(4,)$ float32	100 Hz	EEF orientation $[x, y, z, w]$
`robot0_joint_pos`	$(7,)$ float32	100 Hz	arm joints, rad
`robot0_joint_vel`	$(7,)$ float32	100 Hz	arm joints, rad/s
`robot0_gripper_qpos`	$(2,)$ float32	100 Hz	jaw positions
Contact & tactile			
`force_torque`	$(6,)$ float32	50 Hz	wrist F/T in EEF frame, N + N·m
`force_torque_world`	$(3,)$ float32	50 Hz	wrist force in world frame, gravity-aligned
`robot0_tactile_left`	$(8, 8)$ float32	50 Hz	left finger taxel grid
`robot0_tactile_right`	$(8, 8)$ float32	50 Hz	right finger taxel grid
Action & reward			
`action`	$(7,)$ float32	100 Hz	OSC `cartesian_delta`: $\Delta\text{pos}(3) + \Delta\boldsymbol{\omega}(3) + \text{grip}$
`reward`	scalar float32	100 Hz	Sparse reward

- **Seesaw** (pivot offset ± 0.10 m; beam mass $0.20-0.80$ kg).
- **Hook tools**: J-shape (two variants) and T-shape, parametric handle length and hook radius.
- **Handled cup** (rigid handle, variable curvature).
- **Holed cuboid cage** (one circular hole on a configurable face).
- **Cracker box** (4 size variants, low-friction surface).
- **Plates** (3 thicknesses: 2, 3, 5 mm; 4 diameters).
- **Bowl** (2 sizes).
- **Virtual QWERTY keyboard** with per-key contact triggers and visual feedback.

All custom assets are authored in MJCF, rendered with the default `robosuite` material set, and licensed under CC-BY-4.0. Source meshes are provided in STL formats. We use the YCB-Object meshes, with collision geometry simplified by V-HACD.

I. YCB usage

TABLE VII
YCB OBJECT SUBSETS PER TASK.

YCB ID	Object	Tasks
002	master chef can	pick-place-ycb
003	cracker box	sheltered-grasp
004	sugar box	pick-place-ycb
005	tomato soup can	pick-place-ycb
008	pudding box	sheltered-grasp
010	banana	pick-place-ycb

J. Seeds and rollouts

Each task ships with $S{=}20$ evaluation seeds drawn i.i.d. from the randomization distribution and held out from all demonstration data. Each policy is evaluated with $K{=}5$ rollout repeats per seed (different action-sampling noise but identical reset). Reported success rate is

$$\text{SR} = \frac{1}{SK} \sum_{i=1}^{S} \sum_{j=1}^{K} \mathbf{1}[s_{ij}],$$

K. Compute

Training and evaluation were performed on AWS multi-GPU cloud instances using NVIDIA H100 and L40S GPUs. In total, the project consumed approximately 454 H100 GPU-hours and 1358 L40S GPU-hours across policy training, ablations, and evaluation. Training runs used distributed mixed-precision (BF16) execution. Interactive VLM evaluation additionally incurred substantial API costs, with Stage 3 closed-loop evaluation costing approximately \$1800 in total.

The vision language model (VLM) evaluation pipeline comprises two sequentially executed stages,

each independently assessed. Stage 1 evaluates visual constraint extraction from multi-view scene frames and structured metadata, while Stage 2 assesses high-level plan generation conditioned on the output of Stage 1. Detailed prompt templates for both stages are provided in the supplementary materials.

L. Implementation Details and Agent Design

The Agent operates through a two-stage architectural design that progressively grounds task planning in physical constraints derived from visual observation and sensor data.

In Stage 1, the agent receives multi-view camera frames of the scene alongside task descriptions, robot configuration parameters, and sensor signals (e.g., force and tactile feedback). The agent processes this information to extract physical constraints essential for task completion, such as robot mobility characteristics, motion restrictions imposed by the scene geometry, and robot-specific constraints relevant to the task. The output of Stage 1 is a constraint memory artifact, a structured representation grounding the plan in observed physical reality rather than generic procedural knowledge. This artifact maintains both episodic memory (frame-by-frame observations from each camera view) and semantic memory (facts extracted from observations, such as object contact relationships, functional roles, and critical physical properties). The artifact explicitly documents information that cannot be inferred from visual input alone, indicating what must be confirmed through sensor feedback during execution. This explicit separation of known and unknown information ensures that Stage 2 operates with full awareness of its knowledge boundaries.

In Stage 2, the agent receives the Stage 1 constraint artifact along with the original scene and sensor data. The agent treats all content in the Stage 1 artifact as inviolable constraints that must guide planning. Stage 2 operates through its own reasoning loop, using the extracted constraints and evidence as episodic memory and leveraging the physical facts as semantic memory to establish a grounded understanding of the scene. The agent then reasons about causal and temporal dependencies between actions, using the Stage 1 constraints to establish a partial order over required steps. This reasoning produces an ordered sequence of sub-goals, where each sub-goal specifies: (1) the physical state to be achieved, (2) preconditions that must hold before execution, (3) postconditions expected after completion, and (4) the specific Stage 1 constraint that necessitates this step. The complete sequence of sub-goals constitutes the procedural memory of the task: not merely a collection of facts about the scene, but the actionable knowledge required to execute the task step by step. All intermediate artifacts are preserved: prompts, frames, reasoning traces, validation logs, and explicit constraint links. As multiple task instances are executed, the episodic records accumulate into generalizable semantic knowledge that can be leveraged by future planning instances.

M. Stage 1: Constraint Extraction via ReAct

Stage 1 uses a structured Chain-of-Thought (CoT) prompting strategy. The model receives a single prompt containing multi-view camera frames, task metadata, controller specification, and a prescribed five step reasoning protocol (system instructions) it must execute before producing any output. All reasoning occurs within a single LLM call and gets converted into structured output for stage 2's system instructions.

The reasoning process unfolds in four phases:

1) **Observation:** The agent systematically examines each camera view, reviews sensor data, and ingests the task description.
2) **Reasoning:** The agent asks itself structured questions: What objects are present in the scene? Where are they located? Are there task-specific constraints or tricks imposed by scene configuration?
3) **Constraint Ranking:** Identified constraints are prioritized according to their impact on task success, with task-critical constraints ranked highest.
4) **Verification:** Each constraint claim is validated against actual observed frames or metadata facts to prevent hallucinations or unsupported inferences.

Should the quality check fail, the agent re-executes the stage, with each attempt logged for traceability. The final output is a well-defined constraint memory artifact documenting both episodic observations (what was seen frame-by-frame across cameras) and semantic facts (object relationships, roles, and physical properties).

N. Stage 2: Constrained Plan Generation via ReAct

For Stage 2, the input comprises the constraint artifact from Stage 1 in addition to the original scene frame and the sensor information metadata. The system views all constraints in the artifact from Stage 1 as obligatory conditions on the plan, which none of the sub-goals can violate or ignore.

Stage 2 employs a structured Chain-of-Thought prompt to determine the dependency between actions and causal order through Stage 1 constraints to produce a viable partial order for actions. Prior to generation of sub-goals, the model must first explain why the simplest possible action does not work and which constraint causes its failure. Based on such reasoning,

the model produces a list of sub-goals, specifying their physical outcome, preconditions, post-conditions, and the respective constraint ID from Stage 1. The model also conducts a three-way audit of failure modes to ensure the completion of the plan – F1 (all the necessary prerequisites are included in the plan), F2 (no invalid orders among sub-goals), and F3 (no informational probe sub-goals are discarded). If the combined quality score is too low, the process is repeated up to six times using the original plan as feedback, and the best solution is chosen.

O. Stage 3: Closed-Loop Execution via VLM-Driven ReAct

Stage 3 realises manipulation tasks through a closed-loop **ReAct (Reasoning + Acting)** cycle. At each decision step the VLM receives the full robot and scene state, expressed as structured text and camera images, and selects one motor primitive to execute. The simulator advances until that primitive converges, the resulting state becomes the next observation, and the loop continues until the task succeeds, a budget is exhausted, or the VLM signals completion.

State Representation and Visual Context: At every step, a **structured textual observation summary** is assembled from configurable sensor channels: end-effector pose (position in metres, quaternion, roll/pitch/yaw in degrees), gripper openness, a six-axis wrist force-torque reading in sensor and world frames, per-object names and poses with physical extents (width $\times$depth$\times$ height in metres) resolved from MuJoCo geometry, and camera calibration metadata. This summary is paired with camera images from up to four viewpoints, including an eye-in-hand gripper camera and fixed exocentric cameras, appended as base64-encoded JPEGs.

To provide temporal awareness, a **rolling frame history** of the last H camera batches (default $H = 12$) is prepended to each prompt, oldest first, so the VLM can observe scene evolution rather than a single snapshot. Each additional history batch adds approximately $1{,}800$ input tokens; the rolling window bounds per-step cost independently of episode length.

Prompt Architecture and Working Memory: Every call uses a **slot-based prompt template** that is fully self-contained: task specification, current observation summary, frame history, and working memory are all injected fresh at each step. A fixed block of robot-specific reference documentation covers the world-frame coordinate convention, canonical height landmarks (table: $z \approx 0.82$,m; safe transit: $\sim$1.05,m; grasp/place: 0.82–0.86,m), controller scaling, and common failure modes, giving the VLM the domain knowledge needed for physically plausible planning without fine-tuning.

Because the VLM has no implicit cross-call memory, a **sliding-window working memory** (`ReactMemory`) supplies continuity. After each step an *Observation–Agent* pair is appended (abbreviated state, primitive executed, reward delta), and a window of the last H pairs is serialised into the prompt. The rolling *frame history* thus provides visual continuity while *text memory* provides action continuity.

Action Space and Primitive Vocabulary: The VLM selects from eight **named motor primitives**: MOVEEEF (world-frame position and optional orientation target), CLOSEGRIPPER, OPENGRIPPER, LIFT (vertical offset), DESCEND (lower to target z holding x, y), ATOMICACTION (raw 7D action for N ticks), HOLD, and DONE. MOVEEEF runs a proportional control loop that exits on convergence (position < 5,mm, orientation $< 5°$), so the VLM specifies a step budget rather than a duration. A per-robot **gripper state register** propagates the last commanded open/close state into all movement primitives, preventing mid-transport drops.

VLM Integration and Control Flow: The system supports OpenAI, Anthropic, Google, and AWS Bedrock via a unified interface. Primitives are exposed as typed JSON schemas with `tool_choice: required`, enforcing provider-side schema validation and a structured `reasoning/description/params` response per step. Each call is wrapped in a retry loop (default: 3 attempts); failed attempts execute a HOLD.

The ReAct loop runs for up to K steps (default $K = 24$) and exits on: (1)reward $= 1.0$ (success); (2)DONE signal; (3) simulator termination; or (4) step budget exhausted. A hard simulator-tick cap (default: $10{,}000$ ticks) prevents runaway primitive loops.

P. Per-task VLM Stage-1 and stage-2 results

Per-task results for Stage 1 and Stage 2 are reported in Tables VIII and IX, respectively.

Q. Diffusion Policy

We train Diffusion Policy (DP) from scratch using the LeRobot diffusion policy implementation. Unlike $\pi_{0.5}$ and GR00T N1.5, no pretrained policy checkpoint is finetuned.

R. $\pi_{0.5}$ Finetuning

We finetune the public `lerobot/pi05_base` checkpoint.

S. GR00T N1.5 Finetuning

We finetune `nvidia/GR00T-N1.5-3B` through LeRobot's GR00T policy stack.

TABLE VIII

STAGE-1 UNDERSTANDING ACCURACY PER TASK (%). A RESPONSE IS GRADED CORRECT ON A PROBE IF IT SATISFIES ALL REQUIRED IDENTIFICATIONS AND CONTAINS NO FORBIDDEN FALSE POSITIVES; TASK-LEVEL ACCURACY IS THE MACRO-AVERAGE ACROSS APPLICABLE PROBES

Cat.	Task	GPT-5.4-Mini	Gemma 4	Claude-Haiku 4.5	Claude Sonnet 4.6	GPT-5.5
P1	edge-slide	87.5±4.3	80.0±10.0	**90.0**±0.0	**90.0**±0.0	**90.0**±0.0
P1	recover-peg-insert	43.3±30.9	36.7±26.2	33.3±24.9	43.3±30.9	**80.0**±0.0
P1	sheltered-grasp	90.0±0.0	90.0±0.0	90.0±0.0	90.0±0.0	**100.0**±0.0
P1	cup-inversion	64.0±15.0	74.0±8.0	86.0±4.9	85.0±5.0	**90.0**±0.0
P1	plate-shelf-hobi	84.0±4.9	71.7±6.9	81.7±9.0	**90.0**±0.0	**90.0**±0.0
P1	plate-shelf-hetbi	65.7±11.8	64.3±10.5	76.7±18.9	**90.0**±0.0	**90.0**±0.0
P1	bowl-table	54.0±44.1	34.0±41.8	**92.0**±4.0	82.0±31.2	90.0±0.0
P1	cracker-box-suction	4.0±8.0	4.0±8.0	4.0±8.0	**13.3**±9.4	12.0±24.0
P1	cracker-box-jaw	4.0±8.0	4.0±8.0	**14.0**±28.0	6.0±12.0	0.0±0.0
P2	cube-toss	70.0±20.0	40.0±10.0	80.0±20.0	**95.0**±5.0	90.0±0.0
P2	pendulum-grasp	90.0±0.0	90.0±0.0	90.0±0.0	90.0±0.0	**100.0**±0.0
P2	linear-intercept	**100.0**±0.0	95.0±5.0	95.0±5.0	**100.0**±0.0	90.0±0.0
P2	bounce-intercept	**90.0**±0.0	**90.0**±0.0	**90.0**±0.0	**90.0**±0.0	**90.0**±0.0
P2	galileo-ramp-drop	43.3±9.4	46.7±4.7	65.0±15.0	**85.0**±5.0	66.0±8.0
P2	slide-catch	88.0±14.7	20.0±0.0	89.0±13.6	**95.0**±5.0	80.0±0.0
P2	ramp-sort	32.0±14.7	40.0±12.6	74.0±23.3	**90.0**±0.0	**90.0**±0.0
P2	domino-single	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	90.0±0.0
P3	domino-select	55.0±7.6	44.0±12.0	34.0±10.2	70.0±28.3	**82.0**±7.5
P3	seesaw-balance	83.3±9.4	66.0±13.6	82.5±13.0	**90.0**±0.0	80.0±0.0
P3	indirect-push	**100.0**±0.0	96.0±4.9	97.5±4.3	**100.0**±0.0	**100.0**±0.0
P3	gap-funnel	75.0±5.0	90.0±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0
P3	ball-cage-escape	90.0±0.0	90.0±0.0	90.0±0.0	90.0±0.0	**100.0**±0.0
P3	tool-retrieve	90.0±0.0	90.0±0.0	80.0±10.0	90.0±0.0	**100.0**±0.0
P4	cup-extract	**90.0**±0.0	86.0±8.0	70.0±21.0	**90.0**±0.0	86.0±8.0
P5	mass-sort	**100.0**±0.0	90.0±0.0	**100.0**±0.0	**100.0**±0.0	90.0±0.0
P5	occluder-push	90.0±0.0	90.0±0.0	67.5±39.0	95.0±5.0	**98.3**±3.7
P6	slip-recovery	16.7±12.5	6.7±9.4	30.0±24.5	33.3±24.9	**80.0**±0.0
P6	stack-collapse-recovery	**80.0**±10.0	75.0±15.0	40.0±0.0	55.0±15.0	0.0±0.0
P6	pick-place-ycb	90.0±0.0	70.0±0.0	40.0±0.0	**100.0**±0.0	90.0±10.0
P7	shape-stack	6.7±9.4	23.3±20.5	0.0±0.0	25.0±25.0	**60.0**±11.0
P7	balance-medium	63.3±4.7	43.3±18.9	13.3±9.4	86.7±4.7	**90.0**±0.0
P7	balance-hard	66.7±17.0	33.3±47.1	76.7±20.5	**93.3**±4.7	72.0±9.8
P7	packing	50.0±0.0	50.0±0.0	50.0±0.0	55.0±5.0	**100.0**±0.0
M	mirror-pick-place	0.0±0.0	0.0±0.0	**4.0**±8.0	0.0±0.0	0.0±0.0
M	keyboard-typing	10.0±10.0	0.0±0.0	0.0±0.0	0.0±0.0	**16.7**±23.6

T. Software Environment

Both $\pi_{0.5}$ and GR00T N1.5 are trained in bf16 mixed precision. Flash attention is enabled for GR00T System-1 VLM attention and the $\pi_{0.5}$ PaliGemma backbone.

For each task we list the physical axis the OOD evaluation probes, the training distribution along that axis, and the OOD distribution. OOD distributions are designed to require *extrapolation* of the reasoning to diverse scenarios.

P1 sheltered-grasp (Geometry-aware grasping under occlusion): object sheltered from a canonical lateral side $\rightarrow$ object placed on the opposite side, requiring adaptation of grasp approach

P2 pendulum-grasp (Dynamic interception of a swinging object): pendulum swings in the sagittal plane ($0°$) $\rightarrow$ pendulum swing plane rotated to $45°$, altering timing and approach trajectory

P3 domino-single (Sequential contact-rich manipulation): axis-aligned domino chain configuration $\rightarrow$ domino chain rotated diagonally by $45°$, changing spatial interaction geometry

P4 tool-retrieve (Tool-use for indirect object interaction): standard hook orientation used for retrieval $\rightarrow$ hook mirror-flipped with the tip on the opposite side, requiring strategy adaptation

P5 occluder-push (Manipulation under increasing clutter and occlusion): single occluder lid covering the target $\rightarrow$ 2-3 additional lids stacked on top, increasing obstruction complexity

P6 slip-recovery (Reactive replanning after unexpected disturbances): single slip event during object transport

TABLE IX
STAGE-2 PLAN CORRECTNESS PER TASK (%). PLANS GRADED CORRECT IF A HUMAN VERIFIER FINDS THE GENERATED PLAN SUFFICIENT TO EXECUTE THE TASK CORRECTLY.

Cat.	Task	GPT-5.4-Mini	Gemma 4	Claude-Haiku 4.5	Claude Sonnet 4.6	GPT-5.5
P1	edge-slide	87.5±4.3	80.0±10.0	**90.0**±0.0	**90.0**±0.0	**90.0**±0.0
P1	recover-peg-insert	26.7±18.9	30.0±21.6	26.7±18.9	46.7±33.0	**80.0**±0.0
P1	sheltered-grasp	85.0±5.0	90.0±0.0	90.0±0.0	90.0±0.0	**100.0**±0.0
P1	cup-inversion	34.0±41.8	82.0±4.0	88.0±4.0	40.0±20.0	**90.0**±0.0
P1	plate-shelf-hobi	64.0±10.2	46.7±7.5	43.3±18.0	70.0±8.2	**75.0**±8.7
P1	plate-shelf-hetbi	55.7±7.3	55.7±14.0	76.7±18.9	85.7±4.9	**90.0**±0.0
P1	bowl-table	50.0±41.5	32.0±39.7	**92.0**±4.0	74.0±37.2	88.0±4.0
P1	cracker-box-suction	4.0±8.0	0.0±0.0	0.0±0.0	0.0±0.0	**8.0**±16.0
P1	cracker-box-jaw	0.0±0.0	**4.0**±8.0	**4.0**±8.0	0.0±0.0	0.0±0.0
P2	cube-toss	45.0±45.0	15.0±15.0	80.0±20.0	75.0±15.0	**90.0**±0.0
P2	pendulum-grasp	90.0±0.0	90.0±0.0	90.0±0.0	90.0±0.0	**100.0**±0.0
P2	linear-intercept	**100.0**±0.0	95.0±5.0	95.0±5.0	**100.0**±0.0	90.0±0.0
P2	bounce-intercept	76.7±9.4	70.0±10.0	86.7±4.7	**90.0**±0.0	**90.0**±0.0
P2	galileo-ramp-drop	26.7±4.7	23.3±4.7	40.0±10.0	**75.0**±15.0	30.0±25.3
P2	slide-catch	0.0±0.0	0.0±0.0	15.0±21.4	**95.0**±5.0	0.0±0.0
P2	ramp-sort	26.0±8.0	38.0±14.7	74.0±23.3	**90.0**±0.0	**90.0**±0.0
P2	domino-single	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0	90.0±0.0
P3	domino-select	45.0±12.6	34.0±12.0	26.0±12.0	70.0±28.3	**78.0**±14.7
P3	seesaw-balance	76.7±12.5	42.0±18.3	70.0±12.2	**85.0**±5.0	80.0±0.0
P3	indirect-push	**100.0**±0.0	68.0±39.2	77.5±33.4	**100.0**±0.0	**100.0**±0.0
P3	gap-funnel	90.0±0.0	75.0±15.0	**100.0**±0.0	**100.0**±0.0	**100.0**±0.0
P3	ball-cage-escape	90.0±0.0	90.0±0.0	90.0±0.0	90.0±0.0	**100.0**±0.0
P3	tool-retrieve	60.0±30.0	90.0±0.0	40.0±10.0	60.0±30.0	**100.0**±0.0
P4	cup-extract	**74.0**±10.2	52.0±18.3	24.0±4.9	25.0±5.0	70.0±11.0
P5	mass-sort	95.0±5.0	90.0±0.0	**100.0**±0.0	**100.0**±0.0	90.0±0.0
P5	occluder-push	88.0±4.0	90.0±0.0	67.5±39.0	95.0±5.0	**98.3**±3.7
P6	slip-recovery	6.7±9.4	23.3±20.5	36.7±26.2	36.7±26.2	**80.0**±0.0
P6	stack-collapse-recovery	55.0±15.0	10.0±10.0	**60.0**±0.0	25.0±25.0	0.0±0.0
P6	pick-place-ycb	30.0±0.0	20.0±0.0	50.0±0.0	**90.0**±0.0	**90.0**±10.0
P7	shape-stack	6.7±9.4	6.7±9.4	0.0±0.0	0.0±0.0	**60.0**±11.0
P7	balance-medium	20.0±28.3	6.7±9.4	6.7±9.4	23.3±33.0	**88.0**±4.0
P7	balance-hard	6.7±9.4	33.3±47.1	56.7±41.9	**83.3**±17.0	72.0±9.8
P7	packing	25.0±25.0	0.0±0.0	50.0±0.0	50.0±0.0	**100.0**±0.0
M	mirror-pick-place	0.0±0.0	**4.0**±8.0	**4.0**±8.0	0.0±0.0	0.0±0.0
M	keyboard-typing	10.0±10.0	0.0±0.0	0.0±0.0	0.0±0.0	**16.7**±23.6

→ two independent slips introduced during transport, requiring repeated recovery

P7 balance-medium (Balancing under geometric distribution shifts): limited base and rod orientations during training → both base and rod orientations shifted outside the training distribution

M keyboard-typing (Language-conditioned long-horizon typing): fixed vocabulary words seen during training → previously unseen words requiring compositional generalization

We evaluated performance across the three evaluation stages: constraint understanding, plan generation, and policy execution to identify where competence is gained, lost, or never acquired. Numbers in this section reference Tables VIII, IX, and XV.

U. Stage 1: Physical Constraint Understanding

Stage 1 results show that current VLMs can identify basic physical constraints needed for common manipulation tasks, yet consistently miss fine-grained details that are critical for task success. Models perform relatively well on simpler tasks: domino-single and indirect-push reached 90–100% across all models, linear-intercept scored 95–100%, and bounce-intercept reached 90% uniformly. Performance drops considerably once tasks require precise physical reasoning.

Understands the goal but skips small critical details. Balance and stacking tasks expose this gap clearly. For balance-medium, most models scored in the 43–87% range (Haiku and GPT-5.5 were outliers on this), while shape-stack scores were near floor level (GPT-5.4-Mini 6.7%, Claude-Haiku 0%, Claude-Sonnet 25%), except GPT-5.5 which scored 60%, indicating

TABLE X
STAGE-3 ACTION SUCCESS RATE PER TASK (%) FOR GPT-5.5 WITHOUT AND WITH PRIVILEGED OBJECT CONTEXT.

Cat.	Task	GPT-5.5	GPT-5.5-obj
P1	edge-slide	0.0	0.0
P1	recover-peg-insert	0.0	0.0
P2	bounce-intercept	0.0	0.0
P2	slide-catch	0.0	0.0
P3	domino-single	**100.0**	**100.0**
P3	domino-select	50.0	**100.0**
P4	tool-retrieve	0.0	0.0
P4	cup-extract	0.0	0.0
P5	mass-sort	0.0	0.0
P5	occluder-push	0.0	0.0
P6	slip-recovery	0.0	0.0
P6	pick-place-ycb	0.0	0.0
P7	shape-stack	0.0	0.0
P7	balance-medium	0.0	0.0
M	mirror-pick-place	30.0	**100.0**
M	keyboard-typing	0.0	0.0

TABLE XI
DIFFUSION POLICY TRAINING SETTINGS.

Hyperparameter	Value
Trainable parameters	Trained from scratch
Optimizer	AdamW
Learning rate	1×10^{-4}
LR schedule	Cosine with 500-step warmup
Batch size	32
Training steps	50k steps
Weight decay	1×10^{-6}
Checkpoint save frequency	every 5K steps

TABLE XII
$\pi_{0.5}$ FINETUNING SETTINGS.

Hyperparameter	Value
Base checkpoint	`lerobot/pi05_base`
LoRA config	Rank 64, $\alpha = 128$, dropout 0.05
Observations	4 RGB cameras (front/left/right/-gripper), language instruction
State/action dims	Native single-arm 7D action, bimanual 14D; padded to unified format
Optimizer	AdamW
Learning rate	3×10^{-4}
LR schedule	Cosine decay, 200-step warmup
Batch size	128
Training steps	50K
Gradient clip norm	1.0
Weight decay	0.0

TABLE XIII
GR00T N1.5 FINETUNING SETTINGS.

Hyperparameter	Value
Base checkpoint	`nvidia/GR00T-N1.5-3B`
LoRA config	Rank 64, $\alpha = 128$, dropout 0.05
Observations	4 RGB cameras (resized to 224×224), 15D state, 7D action
Optimizer	AdamW
Learning rate	3×10^{-4}
LR schedule	Cosine decay, 400-step warmup
Batch size	64
Training steps	20K
Gradient clip norm	1.0
Weight decay	0.0

that COM-grounded reasoning is rarely produced for most models. Interestingly, balance-hard showed higher average accuracy than balance-medium for several models—Claude-Haiku improved from 13.3% to 76.7% and Claude-Sonnet from 86.7% to 93.3%—because hard-balance tasks were designed to test sub-point precision, which directly prompted COM positioning and alignment reasoning. In insertion tasks (recover-peg-insert), scores remained moderate and variable (33–80%), consistent with models identifying the high-level insert goal while omitting the required pre-insertion alignment constraint.

Recognizes the object but misses its mechanical role. Cracker-box tasks show the most severe failure in Stage 1. Both cracker-box-jaw and cracker-box-suction scored near zero across all models: jaw scores ranged from 0% (GPT-5.5) to 14% (Claude-Haiku), and suction scores ranged from 4% to 13.3%. Models identified the object and the relevant surface but never produced the specific mechanical constraints required for a valid grasp, such as gripper alignment angle for jaw grasping or correct face selection for suction.

Understands start and end but misses the transition. Orientation tasks such as cup-inversion scored in the 64–90% range, and plate-shelf tasks (hobi/hetbi) scored 65–90%, suggesting that models recognized the target configuration. However, they consistently omitted the rotation or reorientation step required to reach it. Tool-retrieve scores were high (80–100%), meaning models knew the hook was relevant, but as discussed in Stage 2, the hook-to-gripper transition was still missing from their plans.

Detects hidden constraints only when the test is direct. Models handled observable hidden constraints well: occluder-push scored 90–98% across models, and mass-sort reached 90–100%, because both tasks require a single direct action (move the plate, lift the cube) to expose the missing information. In contrast, mirror-pick-place scored 0% for all models except Claude-Haiku at 4%. This task required reasoning that the visual input was horizontally flipped and that left-right motions

TABLE XIV
SOFTWARE ENVIRONMENT USED FOR POLICY TRAINING.

Package	Version
Python	3.12
PyTorch	2.7.1 + cu126
Flash-Attention	2.7.4.post1
robosuite	1.5
MuJoCo	3.1.2
CUDA	12.6

must be reversed, a non-obvious, abstract constraint that no model reliably identified.

Overall, Stage 1 failures follow a consistent pattern: models understand the objective and identify the relevant objects but skip the specific physical constraint that makes the task succeed, **describing tasks procedurally rather than reasoning from physical principles.**

V. Stage 2 Analysis: Constraint-to-Plan Conversion

Since Stage 2 requires grounding Stage 1 understanding into concrete actions, the relatively small performance gap between the two stages suggests that once models understand the constraints, they can usually translate them into appropriate actions. We discuss when the models struggled to convert correctly identified physical constraints from Stage 1 into an ordered and executable action plan.

Mentioned dynamic constraints but did not track them in the plan. The slide-catch task illustrates this gap most sharply. Stage 1 scores were strong for most models (GPT-5.4-Mini 88%, Claude-Haiku 89%, Claude-Sonnet 95%), showing they understood the ramp-tilt dynamics and bin-timing constraint. Yet Stage 2 scores collapsed to 0% for GPT-5.4-Mini, Gemma 4, and GPT-5.5, and only 15% for Claude-Haiku, with Claude-Sonnet the only model maintaining a high score at 95%. Models produced plans that named the bin and the cube but contained no steps for tracking the tilt angle, predicting the release point, or adjusting the bin position dynamically. Knowing the constraint existed was not enough to generate a plan that satisfied it.

Knew a tool switch was needed but did not include it in the plan. In tool-retrieve tasks, Stage 1 scores were uniformly high (80–100% across all models), confirming that the hook-to-gripper transition was identified. However, Stage 2 scores dropped noticeably for several models: Claude-Haiku fell from 80% to 40% and GPT-5.4-Mini from 90% to 60%. Even in cases where the transition appeared in the output, plans often described dragging the hook across the surface without releasing it, which is an incorrect action sequence. This shows that models can recognize a constraint in Stage 1 and still fail to place it correctly in the plan.

Detected slip but did not plan real recovery. Slip-recovery was among the weakest tasks across both stages. Stage 1 scores were already low (GPT-5.4-Mini 16.7%, Gemma 4 6.7%, Claude-Haiku 30%, Claude-Sonnet 33.3%), and Stage 2 scores remained similarly low or declined further (GPT-5.4-Mini 6.7%, Gemma 4 23.3%, Claude-Haiku 36.7%, Claude-Sonnet 36.7%), with GPT-5.5 the only model reaching 80% in both stages. Where slip was mentioned, plans typically added only a monitor or grip-force step. Models planned for observation but not for the corrective action.

Recognized collapse but planned only a partial rebuild. Stack-collapse-recovery showed a consistent downward shift from Stage 1 to Stage 2. GPT-5.4-Mini went from 80% to 55%, Gemma 4 from 75% to 10%, and Claude-Sonnet from 55% to 25%. GPT-5.5 scored 0% in both stages. Stage 2 plans typically described placing the top cube back onto an assumed intact tower, ignoring that all blocks may have scattered and need to be located and restacked from the base upward.

Performed better when Stage 1 constraint was simple geometry. Gap-funnel and sheltered-grasp were the clearest successes. For gap-funnel, Stage 2 scores were high across all models (GPT-5.4-Mini 90%, Claude-Haiku and Claude-Sonnet 100%, GPT-5.5 100%), closely matching their Stage 1 scores. The geometric constraint; align sphere with gap, push through, verify, translated directly into an ordered sequence of actions. Sheltered-grasp similarly maintained high Stage 2 scores (85–100%), because the physical shelter forced a specific entry order that was hard to omit. Ramp-sort also showed good Stage 1 to Stage 2 retention for stronger models, with Claude-Sonnet and GPT-5.5 both at 90% in Stage 2, consistent with their Stage 1 scores.

Sequencing is the core requirement of Stage 2. The majority of Stage 2 errors stemmed from incorrect or incomplete step ordering rather than from missing object references. In tool-retrieve, releasing the hook must come before grasping the cube. The models predicted to use the tool iteself for picking and then releasing the tool. In stack-collapse, locating scattered blocks must precede reconstruction. In slide-catch, tilt tracking and release prediction must precede the catch. If the sequence is wrong, the plan fails even when all objects and constraints are named. Across the board, Stage 2 succeeds when each Stage 1 constraint is converted into a clearly ordered action step, not merely restated in the plan.

W. Stage 3 Analysis: Performance of IMBAgent

Tasks requiring precise geometric alignment showed consistent reasoning failures. In `recover-peg-insert`, the agent failed to identify

TABLE XV
POLICY SUCCESS RATES (%) PER TASK ON IMBENCH. BOLD MARKS THE BEST PER ROW.

		$\pi_{0.5}$		**GR00T 1.5**		**DP**
		ZS	**FT**	**ZS**	**FT**	**Full-training**
T01	edge-slide	0.00	0.07	0.00	0.00	**0.28**
T02	recover-peg-insert	0.00	0.03	0.00	0.00	**0.29**
T03	sheltered-grasp	0.00	0.10	0.00	0.00	**0.30**
T04	cup-inversion	0.00	0.00	0.00	0.00	0.00
T05	plate-shelf-hobi	0.00	0.00	0.00	0.00	0.00
T06	plate-shelf-hetbi	0.00	0.00	0.00	0.00	0.00
T07	bowl-table	0.00	**0.27**	0.00	0.00	0.00
T08	cracker-box-suction	0.00	0.00	0.00	0.00	0.00
T09	cracker-box-jaw	0.00	0.00	0.00	0.00	0.00
T10	cube-toss	0.00	**0.12**	0.00	0.00	0.09
T11	pendulum-grasp	0.08	0.28	0.00	0.08	**0.35**
T12	linear-intercept	0.00	**0.04**	0.00	**0.04**	0.01
T13	bounce-intercept	0.00	0.00	0.00	**0.04**	0.00
T14	galileo-ramp-drop	0.00	0.00	0.00	0.00	**0.34**
T15	slide-catch	0.00	**0.10**	0.00	0.00	0.00
T16	ramp-sort	0.00	0.00	0.00	0.00	**0.07**
T17	domino-single	0.28	**0.73**	0.08	0.25	0.70
T18	domino-select	0.00	0.32	0.08	0.17	**0.80**
T19	seesaw-balance	0.00	0.10	0.00	0.00	**0.13**
T20	indirect-push	0.00	0.07	0.00	0.00	**0.08**
T21	gap-funnel	0.00	0.06	0.00	0.00	**0.07**
T22	ball-cage-escape	0.22	**0.45**	0.00	0.22	0.43
T23	tool-retrieve	**0.18**	0.00	0.00	0.00	0.00
T24	cup-extract	0.00	0.02	0.00	0.00	**0.07**
T25	mass-sort	0.00	0.00	0.00	0.00	0.00
T26	occluder-push	0.00	0.15	0.00	0.00	**0.50**
T27	slip-recovery	0.00	0.22	0.00	0.00	**0.44**
T28	stack-collapse-recovery	0.00	0.00	0.00	0.00	0.00
T29	pick-place-ycb	0.00	**0.30**	0.00	0.00	0.28
T30	shape-stack	0.00	0.00	0.00	0.00	0.00
T31	balance-medium	0.00	**0.71**	0.00	0.02	0.28
T32	balance-hard	0.00	**0.58**	0.00	0.03	0.20
T33	packing	0.00	0.00	0.00	0.00	0.00
M01	mirror-pick-place	0.02	0.00	0.00	0.00	**0.98**
M02	keyboard-typing	0.00	0.13	0.00	0.04	**0.32**

the correct insertion axis and often rotated in the wrong direction even with privileged object-centric information (GPT-5.5 0%, GPT-5.5-obj 0%). The agent confused the hole orientation and never achieved alignment. Similarly, in `balance-medium`, the agent partially understood the task by grasping near the rod's center, but reduced execution to a simple pick-and-place behavior instead of true balancing (GPT-5.5 0%, GPT-5.5-obj 0%). Failed placements were never corrected or retried, showing limited online recovery.

Tasks that only required approximate interactions showed strong performance. In `domino-single`, the agent repeatedly attempted corrective pushes after failures and eventually succeeded in collapsing the domino chain in all runs (GPT-5.5 100%, GPT-5.5-obj 100%). A similar trend appeared in `domino-select`, where privileged object-centric information removed ambiguity about the target chain (GPT-5.5 50%, GPT-5.5-obj 100%). With accurate pose information, the agent reliably aligned with the correct domino sequence and achieved high success rates.

Across several tasks, policies followed open-loop plans and rarely adapted after execution errors. In `pick-place-ycb`, the agent continued executing the initial plan even after failed grasps or incorrect placements, often treating the task as complete regardless of the final outcome (GPT-5.5 0%, GPT-5.5-obj 0%). These failures were amplified by irregular YCB object geometries, where unstable grasps required corrective adjustments that never emerged.

Privileged information can both help performance and hurt reasoning. Object-centric information improved performance in tasks where ambiguity was mainly spatial, but hurt reasoning about transformations or hidden structure. In `domino-select`, exact pose

information enabled reliable target identification and directional pushing (GPT-5.5 50%, GPT-5.5-obj 100%). In contrast, `mirror-pick-place` exposed failures in reasoning about mirrored coordinate frames (GPT-5.5 30%, GPT-5.5-obj 100%). In image-only settings, the agent sometimes succeeded by visually grounding the mirrored scene and correcting actions online. However, with privileged poses, the agent ignored image observations and directly executed actions toward the reported coordinates without transforming them into the mirrored frame.

Policies fail to infer latent interaction strategies. Several failures came not from low-level control, but from failing to infer the correct physical strategy. In `cup-extract`, the agent never discovered that rotating the cup was necessary to release the cube inside (GPT-5.5 0%, GPT-5.5-obj 0%). Instead, it often grasped the rim rather than the handle, blocking the object from exiting. Similarly, in `occluder-push`, the agent often failed to infer that the cube could be hidden beneath the plates and prematurely terminated execution, treating the task as impossible (GPT-5.5 0%, GPT-5.5-obj 0%). Even when the plates were identified as occluders, the generated actions failed to remove them effectively. With privileged object poses, the agent bypassed occluder reasoning entirely and attempted direct interaction with the hidden target, again resulting in failure.

X. End-to-end Visuomotor Policies

We discuss task-specific behaviors, emergent properties of the policies, their success and failure modes, implications of these behaviors, and the physical reasoning axes that current policies fail to capture. All values in the section below are referred from the table T for the Finetuned values.

a) Coarse-contact tasks (domino-select, domino-single, seesaw-balance).: Tasks that require only coarse directional interaction, rather than precise end-effector placement, achieve the highest success rates across all evaluated policies (*domino-single*: $\pi_{0.5}$ 73%, GR00T 25%, DP 70%; *domino-select*: $\pi_{0.5}$ 32%, GR00T 17%, DP 80%). We observed that even approximately correct behaviors can solve the domino tasks, sometimes by directly knocking intermediate dominoes instead of propagating through the full chain. A similar pattern appears in *domino-select*, where policies learn an approximate spatial correspondence for selecting the correct chain. The motion remains coarse, but the correspondence is learned reliably. On *seesaw-balance*, policies failed to exhibit emergent corrective behavior ($\pi_{0.5}$ 10%, GR00T 0%, DP 13%). Unlike humans in teleoperation demonstrations, the policies did not repeatedly adjust load placement until the beam reached equilibrium.

For tasks requiring accurate end-effector placement within a narrow success region, policies often demonstrate qualitatively correct motion plans but fail at the geometry level. On ***balance-hard***, the object is typically grasped successfully and placed near the target location ($\pi_{0.5}$ 58%, GR00T 3%, DP 20%). Importantly, the model learns to grasp near the center of mass, which is crucial for stable manipulation. The policy also shows adjustment behaviors to better align the center of mass during placement. However, the harder balance variants require substantially more precise control, leading to lower success rates (balance-medium: $\pi_{0.5}$ 71%, GR00T 2%, DP 28%). In ***ball-cage-escape***, despite moderate task success ($\pi_{0.5}$ 45%, GR00T 22%, DP 43%), the policy behavior differs significantly from human strategies. Human teleoperators first align the cage into orientations from which rotating the object naturally releases the ball. Policies fail to consistently learn this strategy, although they still perform multiple reorientation attempts, some of which succeed. On ***keyboard-typing***, the high-level intent is preserved, but cumulative end-effector drift causes frequent key misalignments ($\pi_{0.5}$ 13%, GR00T 4%, DP 32%). The correct finger posture often reaches an adjacent key instead of the target. Furthermore, in nearly 60% of failures, the policy misunderstands the task semantics and presses keys that were already pressed.

b) Dynamic interception and predictive tasks (bounce-intercept, linear-intercept, pendulum-grasp).: Policies generate trajectories that approximately follow the motion direction of the target object but fail to accurately match its speed (bounce-intercept: $\pi_{0.5}$ 0%, GR00T 4%, DP 0%; linear-intercept: $\pi_{0.5}$ 4%, GR00T 4%, DP 1%). An interesting observation is that successful episodes primarily occur when the object moves toward the robot arm, while failures are common when the object moves away. In *pendulum-grasp*, the policy learns to move toward a small set of likely grasp locations due to the limited number of feasible interception points ($\pi_{0.5}$ 28%, GR00T 8%, DP 35%). Common failure modes include grasping too early or too late, indicating partial understanding of timing, proximity, and reactivity.

The ***edge-slide*** task fails for most VLAs with a similar failure mode: the policies apply excessive force while exposing the edge, causing the object to fall ($\pi_{0.5}$ 7%, GR00T 0%). Diffusion Policy learns smoother motion speeds compared to VLAs, but the dominant failure mode remains similar (DP 28%).

c) Mirror correspondence and adaptation (mirror-pick-place).: Mirror-pick-place results in near-zero performance for most VLAs ($\pi_{0.5}$ 0%, GR00T 0%). This strongly supports our central hypothesis that

humans possess implicit intuitive correspondence reasoning. Human teleoperators were shown only mirrored observations and, after roughly 20–30 trials, adapted reliably to the reversed control mapping. Humans quickly infer correspondences such as "left becomes right" and "up becomes down."

This adaptation does not emerge in VLAs, likely because mirrored interaction is outside the distribution of their pretraining data. In contrast, Diffusion Policy performs substantially better, achieving nearly 98% success (DP 98%), likely because it learns representations directly from task-specific demonstrations. Interestingly, our Stage 3 IMBAgent achieves approximately 30% success and occasionally discovers through iterative interaction that observation-action correspondences are reversed (GPT-5.5 30%, GPT-5.5-obj 100%). These results highlight that our benchmark evaluates not only physical understanding but also broader forms of intuitive reasoning.

d) Compositional organization and packing (packing, ramp-sort).: In *packing*, none of the evaluated policies achieve successful completion ($\pi_{0.5}$ 0%, GR00T 0%, DP 0%). Policies attempt pick-and-place behaviors but fail to organize subtasks in a way that satisfies the geometric packing constraints. The objects fit together in only a small set of valid configurations, enforced by the L-shaped geometry. In *ramp-sort*, we observe two dominant failure modes ($\pi_{0.5}$ 0%, GR00T 0%, DP 7%). First, policies often fail to place the ball on the correct side of the ramp. Second, even when placement occurs on the correct side, inaccurate positioning causes the ball to roll into the wrong bin. As a result, the task is highly sensitive to action precision.

e) Bimanual coordination (hetbi-plate-shelf, hetbi-bowl-table).: The heterogeneous bimanual tasks are among the most challenging tasks in the benchmark, as they require simultaneous success across multiple coordination axes (hetbi-plate-shelf: $\pi_{0.5}$ 0%, GR00T 0%, DP 0%). Failures are distributed across several stages of execution. Grasping itself is usually reliable, suggesting that finetuning successfully teaches the policies to acquire the object. Approximately 80% of failures occur during the handover stage. Interestingly, a similar trend is observed in the bowl-table task ($\pi_{0.5}$ 27%, GR00T 0%, DP 0%), although some failed handovers accidentally invert the bowl and still lead to successful upright placement on the table. In such cases, the task semantics are not followed strictly, but the final goal state is nevertheless achieved.

f) Physical stability reasoning (shape-stack).: In *shape-stack*, we observe that the notion of physical stability is not naturally encoded in current policies ($\pi_{0.5}$ 0%, GR00T 0%, DP 0%). Human teleoperators quickly realize that certain objects, such as cones, cannot produce stable stacks unless placed on top. Policies learn the stacking action itself but fail to reason about long-term physical stability.

g) Active discovery under occlusion (occluder-push).: Diffusion Policy significantly outperforms VLAs on *occluder-push* ($\pi_{0.5}$ 15%, GR00T 0%, DP 50%). VLA baselines frequently attempt to directly pick the occluding plates instead of performing clearing motions for active discovery. Diffusion Policy, trained from scratch on demonstrations, learns the intended clearing behavior more effectively. The dominant failure mode for Diffusion Policy occurs when the clearing motion becomes too aggressive, causing both the occluding plates and the hidden cubes to fall off the table simultaneously.

Overall, the rollout analysis reveals a consistent gap between high-level intent and physically precise execution. Policies often learn coarse task structure, approximate correspondences, and object interaction strategies, but fail when tasks require accurate geometry, stability reasoning, predictive timing, or adaptive corrections. Success is strongest in tasks with tolerant dynamics and broad success regions, while failures are dominated by incorrect physical reasoning, and limited adaptation where intuitive actions are required.

h) Hardware.: Training and evaluations reported in §IV ran on a mix of NVIDIA H100-80GB GPUs (for VLAs), and on L40S (48 GB) for Diffusion policy.

i) Vision-Language Models APi versions: Evaluations used the following VLM backbone versions:

- **GPT-5.5** (`gpt-5.5-2026-04-23`)
- **GPT-5.4-mini** (`gpt-5.4-mini-2026-03-17`)
- **Claude Sonnet 4.6** (`claude-sonnet-4-6`)
- **Claude Haiku 4.5** (`claude-haiku-4-5-20251001`)
- **Gemma 4 31B Instruct** (`google/gemma-4-31B-it`)

All models were queried via their respective APIs or hosted inference endpoints. No fine-tuning was applied to any VLM backbone. All the models were used with maximum reasoning.

j) Data:

- Sample Data: Hosted on Hugging Face Datasets. Individual task datasets can be explored by entering the dataset ID in the official Hugging Face dataset viewer.