CAPEK 0.5: AN EXECUTION-CENTRIC VISION-LANGUAGE MODEL FOR EMBODIED INTELLIGENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language models are increasingly serving as the reasoning core of embodied agents. Robot execution is inherently iterative: each action reshapes the scene and physical state, continually renewing what must be perceived, reasoned about, and verified. Meeting these demands requires complementary capabilities that differ in supervision signals, prediction formats, and verification criteria. Existing approaches typically develop these capabilities against isolated, task-specific objectives, leaving open how they should be organized and integrated around execution as a whole. We present Capek 0.5, an embodied vision-language model built around an execution-centric capability taxonomy. Rather than organizing training by datasets or tasks, the taxonomy groups embodied capabilities according to their functional roles throughout execution and comprises four capability families: Spatial Reasoning, Temporal Understanding, Action Guidance, and State Verification. Each capability is first acquired by a dedicated specialist through reinforcement learning with verifiable rewards from a shared backbone, and the specialists are then consolidated into a single inference-time model through weight-space merging followed by routed policy-space distillation. We evaluate Capek 0.5 from three complementary perspectives: comprehensive benchmark suites including Capek-StateBench, a new benchmark for state verification; a controlled study of capability retention from specialists to the unified model; and closed-loop evaluation in simulated embodied environments. Capek 0.5 improves the large majority of matched benchmark rows over its initialization, retains all four specialized capabilities in one checkpoint with quantified losses, and transfers to closed-loop embodied task execution. The results indicate that execution-centric specialization followed by consolidation is a practical recipe for building unified embodied VLMs.

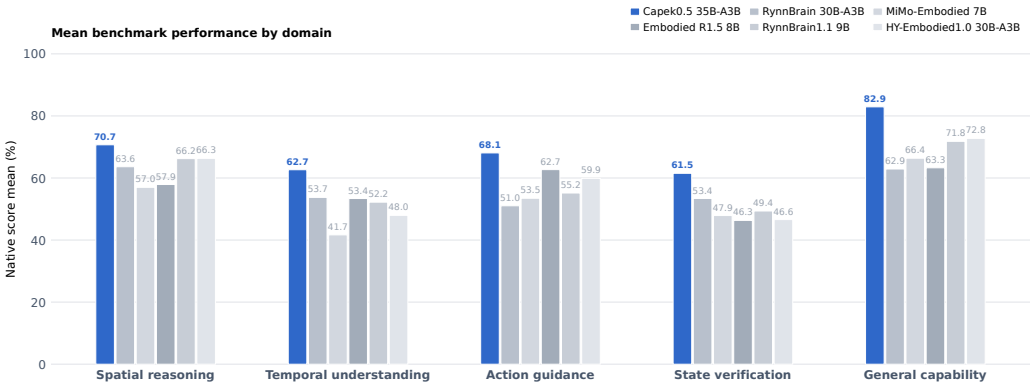


Figure 1: **Average performance across capability domain.** Results from Table 1 are averaged within each domain. NaviTrace, VABench-trace, and ShareRobot-Trajectory are excluded from the Action Guidance average due to incompatible trajectory metrics. The results show that Capek 0.5-35B-A3B maintains strong and balanced performance across embodied and general capabilities.

1 INTRODUCTION

Vision-language pretraining supplies strong perceptual and linguistic priors (Radford et al., 2021; Alayrac et al., 2022; Li et al., 2023b; Dai et al., 2023; Liu et al., 2023a; Wang et al., 2024a; Bai et al., 2025b), enabling vision-language models (VLMs) to support semantic planning, visual reasoning,

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

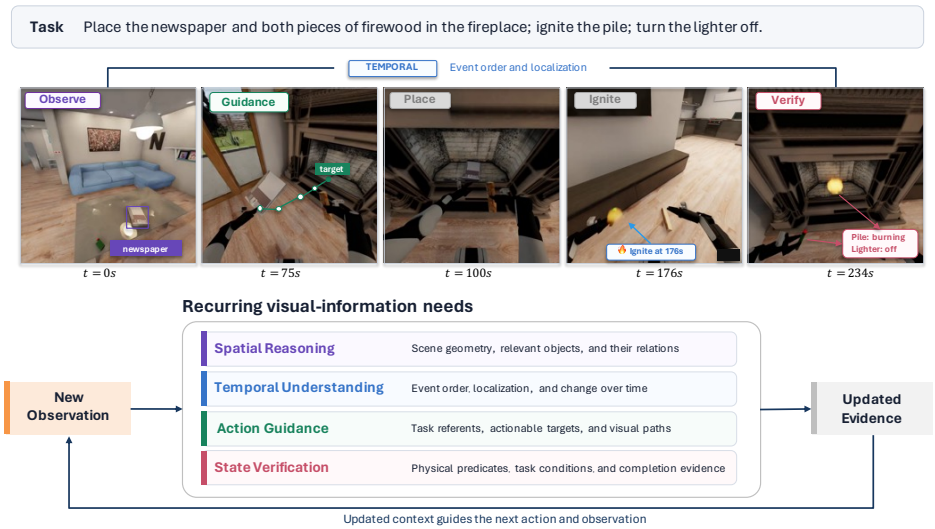


Figure 2: **Execution-centric capability view.** A BEHAVIOR-1K household manipulation trace (Li et al., 2023a) illustrates how Spatial, Temporal, Guidance, and State Verification requirements recur and overlap across an observation–reasoning–action cycle rather than form mutually exclusive stages.

and language-based interfaces to downstream robot control (Ahn et al., 2022; Driess et al., 2023; Liang et al., 2022; Huang et al., 2023; Gemini Robotics Team, 2025). Recent robot-centric models further extend this foundation with complementary abilities such as affordance grounding, trajectory prediction, temporal understanding, and physically grounded planning (Yuan et al., 2024; Ji et al., 2025; Tan et al., 2026; Dang et al., 2026; Wang et al., 2026c; NVIDIA, 2026; Hao et al., 2025b; Yuan et al., 2026; Bjorck et al., 2026; ACE-Brain Team et al., 2026), positioning VLMs as an "embodied brain" that perceives, reasons, and directs action for increasingly capable embodied agents (Yang et al., 2025b; Salimpour et al., 2025).

Yet current models still face challenges because robot execution is inherently iterative: each action changes the scene and physical state, continually renewing the evidence required for subsequent decisions. Throughout this *execution cycle*, a VLM must repeatedly reason about spatial context, interpret events over time, ground actions to the environment, and determine whether physical predicates hold and whether task goals have been achieved. Effective robot execution therefore relies on intermediate predictions that remain interpretable and verifiable, allowing each decision to reliably inform the next. Existing post-training pipelines are commonly organized around individual tasks, datasets, or benchmark-specific objectives. Even when such supervision is mixed within a single model, heterogeneous output structures and scoring rules make it difficult to determine which execution capabilities have been acquired and whether they are retained after integration.

In this work, we present Capek 0.5, a unified VLM for robot perception and reasoning. Rather than organizing post-training solely around individual tasks or datasets, Capek 0.5 adopts an *execution-centric* capability taxonomy that groups capabilities according to their functional roles throughout robot execution. This taxonomy provides a common framework for training and evaluating capabilities that are repeatedly invoked as the robot perceives, acts, and verifies progress in dynamic environments. Concretely, it comprises four capability families: **Spatial Reasoning** captures scene geometry and inter-entity relations; **Temporal Understanding** models event progression and identifies when relevant changes occur; **Action Guidance** grounds task-relevant entities and affordances and produces image-space points, regions, and ordered trajectories; and **State Verification** determines whether visually supported physical predicates hold and the task goals have been achieved. Figure 2 illustrates how the capabilities recur throughout a household manipulation episode as execution proceeds.

Unifying this taxonomy within a single model is nontrivial: the four capabilities differ sharply in their supervision signals and output formats, spanning structured relations, temporal localization, geometric grounding, and visually grounded predicates. Such heterogeneous objectives make joint post-training with a single undifferentiated recipe susceptible to optimization interference across capabilities (Ma et al., 2026; Wang et al., 2026b). To address this challenge, we separate capability acquisition from capability integration. All four specialists start from the same backbone checkpoint with capability-aligned data and objectives. Specialist checkpoints are then combined with TIES (Yadav et al., 2023)

to initialize a unified student, which is further consolidated through routed multi-teacher on-policy distillation (MOPD) (Ma et al., 2026) on student-generated prefixes.

We instantiate two model scales at 35B-A3B and 2B, and evaluate their performance from three complementary perspectives. First, we assess both general and embodied reasoning capabilities by comparing the two model tracks against scale-matched baselines on a comprehensive suite of benchmarks. Second, we conduct a controlled consolidation analysis comparing the pretrained backbone (i.e., Qwen3.6-35B-A3B), four standalone specialists, and different model-merging variants to measure capability retention and consolidation effectiveness. Third, beyond isolated benchmarks, we evaluate Capek 0.5 in simulated embodied environments, including EmbodiedBench (Yang et al., 2025b) and VIGIL (Chen et al., 2026), to assess how diverse reasoning capabilities compose during task execution. In summary, our main contributions are:

- We introduce an execution-centric capability taxonomy together with typed, verifiable task formats and same-origin capability specialists for developing capability acquisition and retention.
- We present Capek 0.5, a unified embodied reasoning model family that consolidates capability specialists into a single model checkpoint through TIES initialization followed by routed MOPD.
- We provide a multi-level evaluation spanning different perspectives to demonstrate model effectiveness. Capek 0.5-35B-A3B and Capek 0.5-2B improve 28 of 34 and 30 of 34 matched benchmark rows against their protocol-matched Qwen initializations.

2 Execution-Centric Capabilities and Supervision

We organize embodied reasoning around four complementary capabilities that recur throughout robot execution: **Spatial Reasoning** captures scene geometry and inter-object relations; **Temporal Understanding** interprets events and changes over time; **Action Guidance** grounds instructions into visual targets, affordances, and trajectories; and **State Verification** assesses physical predicates and task progress. Together, these capability families cover recurring information needs across a broad range of embodied tasks.

We translate this taxonomy into capability-aligned supervision by assigning each sample to a specialist according to its primary execution role. Each specialist supports multiple output formats, unified through shared serialization conventions while retaining their target semantics and verification criteria. Figure 3 summarizes the training data. Detailed data sources and samples are provided in Appendix B.

2.1 Capability-Aligned Supervision

We construct supervision for each capability according to its role in embodied execution. Below, we outline the capabilities and their supervision forms; detailed data sources, construction procedures, and standardization are provided in Appendix B.

Spatial Reasoning. Spatial reasoning concerns the perception of scene geometry and the relations among entities, the observer, and different viewpoints (Liu et al., 2025). It supports embodied decision-making, including determining where one object lies relative to another, estimating target distance and scale, assessing placement feasibility, and reasoning about how spatial relations transform across viewpoints. Our Spatial supervision combines single-view geometry with reasoning across observations, covering direction and orientation, depth and distance, relative size, support and containment, placement, and cross-view correspondence.

Temporal Understanding Temporal understanding enables an embodied agent to reason about dynamic visual evidence: what happens, in what order, when it occurs, and how the scene changes over time. Unlike static-image understanding, robot execution requires the model to track event progression, causal relationships, temporal boundaries, and action-conditioned changes across observations. Accordingly, our Temporal supervision combines video reasoning with temporal localization, requiring the model both to interpret events distributed over time and to identify when they occur.

Action Guidance. Action Guidance connects instructions to actionable visual targets that support downstream execution. The derived questions are produced through deterministic transformations, and their targets are recomputed from the underlying geometric annotations. We therefore unify visual grounding, pointing, affordance localization, and trajectory prediction within a single capability family because they all produce action-oriented visual guidance despite differing prediction formats.

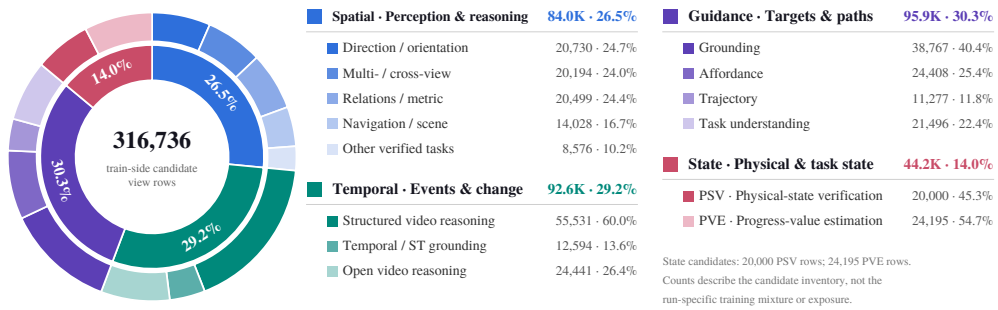


Figure 3: **Training-data pool.** Inner ring: capability families; outer ring: supervision products. Counts are train-side candidate view rows.

State Verification. State verification assesses whether the visually observed execution state supports the conditions required to continue or terminate a robot task. Compared with other embodied reasoning capabilities, verifying execution outcomes remains underexplored. To address this gap, we propose a new verification task spanning two execution-aligned levels. At the *physical level*, the model determines whether a recent action has produced the required object state or relation. At the *task level*, it estimates how much of the high-level instruction has been completed and what should happen next. We construct supervision from execution traces at both levels, as detailed in Section 2.2.

2.2 State Verification: A New Dual-Level Formulation

We introduce a dual-level formulation of State Verification that jointly addresses execution outcomes and task progress, together with supervision constructed with execution traces from BEHAVIOR-1K household-manipulation Li et al. (2023a). Rather than sampling observations uniformly, we anchor examples at execution-relevant primitive, skill, and task checkpoints. Task hierarchies, transition records, official BDDL goal conditions, simulator states, action histories, and future observations may be used on the teacher side to select checkpoints and derive or audit supervision targets. The model receives only the instruction and the causal visual evidence available at the selected checkpoint.

We materialize two model-facing supervision types. At the *physical level*, *physical-state verification (PSV)* uses minimal local causal evidence to determine whether an object state or relation satisfies the requirement of the current primitive or skill. The retained evidence may be a single critical frame, an aligned frame list, or a short transition clip, and covers states and relations such as open or closed, powered on or off, grasp and release, containment, support, and other action-result predicates.

At the *task level*, *progress-value estimation (PVE)* uses the causal visual task prefix to estimate how much of the instruction has been completed and, for paired examples, predict the immediate annotated next action. Each record explicitly identifies whether progress is defined by the fraction of satisfied BEHAVIOR-1K goal conditions or by the fraction of completed primitive child skills, including navigation; the two progress definitions remain separate label spaces. Progress-only and progress-plus-next-step records are treated as output variants of the same PVE supervision type. The underlying goal-condition or child-skill decomposition is used to define these targets but is not itself exposed as an output. In this way, PVE ties progress and next-step supervision to the component structure of the task without exposing symbolic execution traces or future observations.

Before training, samples are filtered for visual support, annotation consistency, and duplication. Figure 4 illustrates how a single trajectory is materialized into local physical-state evidence and shared causal task-state evidence.

3 Model Training

Capek 0.5 builds on the Qwen vision-language series (Qwen Team, 2026b;a) and comprises two scales: 35B-A3B and 2B. It adopts a two-stage post-training paradigm that separates capability acquisition from capability consolidation. In the first stage, four capability specialists are independently optimized from the same backbone checkpoint. In the second stage, the specialists are consolidated into a single model through weight-space merging followed by on-policy distillation, as summarized in Figure 5.

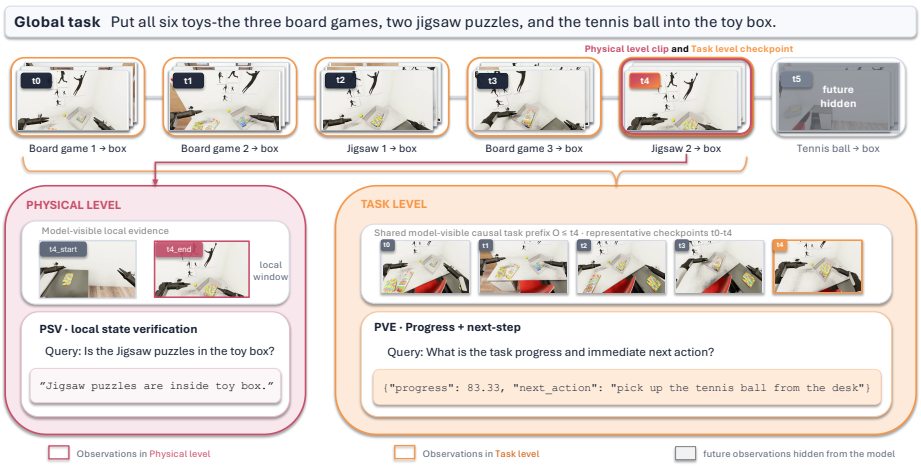


Figure 4: **Execution-centric construction of State Verification supervision.** Starting from one BEHAVIOR-1K trajectory and its global task instruction, we retain checkpoint-aligned evidence segments at two semantic levels. Physical-state verification uses a local transition window to verify an action-result predicate, whereas task-level supervision uses the causal observation prefix through the current checkpoint. The shared task prefix supports both explicit goal-condition verification and the compact progress-value estimation readout. Future observations are hidden from the model.

3.1 Capability Specialized Training

In the capability specialization stage, all specialists are independently optimized from a shared backbone under a unified reward formulation with task-specific verifiers using Group Relative Policy Optimization (GRPO), which keeps the resulting checkpoints parameter-compatible for consolidation.

All specialists are optimized with GRPO (Shao et al., 2024; Guo et al., 2025). For a input–target pair $(x, y^\star)$, the rollout policy $\pi_{\theta_{\text{old}}}$ samples a group of G responses $\{y_i\}_{i=1}^{G}$, each scored by its format-specific reward $r_i = R(y_i, y^\star)$. GRPO computes a response-level advantage by normalizing rewards within the group, $\hat{A}_i = \frac{r_i - \bar{r}}{\sigma_r + \delta}$, where $\bar{r} = G^{-1}\sum_i r_i$ and σ_r are the group mean and standard deviation of the rewards. The policy is updated by the token-level clipped objective

$$\mathcal{L}_{\text{RL}}(\theta) = -\frac{1}{\sum_{i=1}^{G}|y_i|}\sum_{i=1}^{G}\sum_{t=1}^{|y_i|}\Big[\min\big(\rho_{i,t}\,\hat{A}_i,\ \text{clip}(\rho_{i,t}, 1-\epsilon, 1+\epsilon)\,\hat{A}_i\big) - \beta\,\widehat{D}_{\text{KL},i,t}\Big], \quad (1)$$

where $\rho_{i,t}$ is the importance ratio, ϵ is the clipping threshold, and $\beta \geq 0$ is the KL coefficient. Although the four specialists produce heterogeneous outputs, they all share the same reward structure: an optional format reward and a task-accuracy reward. The format reward checks whether the output is well-formed and parseable (e.g., a bounding box has four ordered coordinates) while the accuracy reward measures correctness against the target. The total reward is their weighted sum,

$$R(\hat{y}, y^\star) = \lambda_{\text{fmt}}\, R_{\text{fmt}}(\hat{y}) + \lambda_{\text{acc}}\, R_{\text{acc}}(\hat{y}, y^\star), \quad (2)$$

where $R_{\text{fmt}} \in \{0, 1\}$ is the format reward and $R_{\text{acc}} \in [0, 1]$ is the accuracy score, computed by a rule-based verifier or, for open-ended outputs, a frozen semantic judge. The nonnegative weights sum to one, so $R \in [0, 1]$. Table 5 lists the output formats and their scoring rules. Details of reward design for each capability are provided in Appendix D.0.1.

3.2 Capability Consolidation

After capability specialization, the four experts are consolidated into a single model through a two-step procedure. We first perform weight-space merging using TIES to obtain a unified initialization, which combines complementary parameter updates while mitigating conflicting weight changes. Starting from this initialization, routed Multi-Teacher On-Policy Distillation (MOPD) further transfers the specialists' behaviors into a single policy by supervising student-generated trajectories.

Weight-Space Initialization. TIES provides a conflict-aware weight-space initialization for the unified student. Concretely, let θ_0 be the shared backbone and θ_e the checkpoint of specialist e.

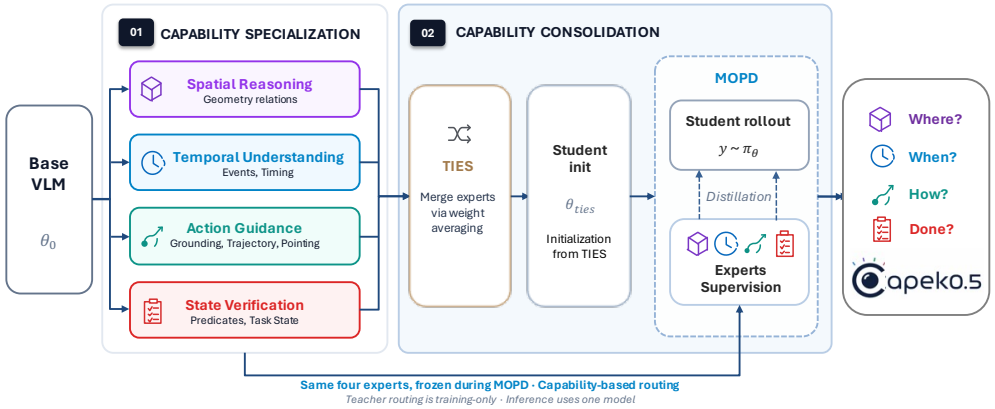


Figure 5: Capability specialization and consolidation training process.

Since all specialists share the same architecture, their parameter updates are directly compatible: $\Delta_e = \theta_e - \theta_0$. We use TIES (Yadav et al., 2023; Ilharco et al., 2023), which resolves conflicts among specialist updates by trimming low-magnitude parameters: $\theta_{\text{TIES}} = \theta_0 + \lambda\ \text{TIES}_\tau\big(\{\Delta_e\}_{e=1}^{E}\big)$, where $E = 4$, $\tau = 0.8$ masks the 80% lowest-magnitude entries within each task vector, and $\lambda = 1.0$ is the merged-update scale. The resulting checkpoint θ_{TIES} initializes the subsequent routed MOPD.

Routed MOPD. Parameter merging alone does not guarantee that every specialist behavior is preserved in the merged model. We therefore further refine the TIES-initialized student using routed MOPD (Ma et al., 2026; Agarwal et al., 2023). During training, each sample is routed to its corresponding specialist teacher, which supervises the student on its own generated trajectories. This allows each capability to be distilled directly from the responsible specialist.

For an input x, the student first samples a response $y \sim \pi_\theta(\cdot \mid x)$, and the routed teacher q_r is then evaluated on the resulting student-generated prefixes $h_t = (x, y_{<t})$, where r indexes the capability route selected for the sample. The student is optimized to minimize the token-level reverse KL to the routed teacher: $\mathcal{L}_{\text{MOPD}}(\theta) = \mathbb{E}\Big[\frac{1}{|y|}\sum_t D_{\text{KL}}(\pi_\theta(\cdot \mid h_t) \,\|\, q_r(\cdot \mid h_t))\Big]$. Throughout training, only the routed teacher is queried for each sample, and all specialist teachers remain frozen. In practice, rather than evaluating the full-vocabulary KL, we optimize a sampled surrogate: the teacher supplies the log-probability of each student-sampled token, which defines a per-token distillation advantage

$$\hat{A}_t^{\text{MOPD}} = \text{clip}\left(\text{sg}\left[\log \frac{q_r(y_t \mid h_t)}{\pi_\theta(y_t \mid h_t)}\right], -\epsilon_{\max}, \epsilon_{\max}\right), \tag{3}$$

where $\text{sg}[\cdot]$ is the stop-gradient operator. Since rollouts are generated on-policy from π_θ, this advantage is optimized directly with the token-mean policy-gradient objective.

4 Experiments and Analysis

We evaluate Capek 0.5 on a comprehensive suite of general and embodied benchmarks spanning Spatial Reasoning, Temporal Understanding, Action Guidance, and State Verification. Since existing benchmarks do not assess whether a model can verify the physical world state after execution or track progress toward task completion, we introduce **Capek-StateBench**, a benchmark with two complementary tracks: **Capek-StateBench-P** for physical-state verification and **Capek-StateBench-T** for task-state verification. The evaluation protocol is described in Appendix C, and the experimental settings are detailed in Appendix D.

4.1 Overall Benchmark Results

We treat the 35B-A3B track in Table 1 as the primary comparison and use the 2B track in Table 8 to check whether the same training recipe yields a similar profile at smaller capacity. Within each table, the protocol-matched Capek 0.5–Qwen pair is the main evidence for post-training effects; external models provide scale-aware context. Because the rows preserve native task formats, including accuracy, mean IoU, point F1, normalized trajectory score, RMSE, and DFD, we analyze directional row-wise changes rather than averaging heterogeneous metrics.

Table 1: Benchmark results for the 35B-A3B variant, compared with A3B MoE and 7–9B dense models of comparable active compute.

Task family	Benchmark	Qwen3.6 35B-A3B	**Capek0.5 35B-A3B**	RynnBrain 30B-A3B	MiMo-Emb. 7B	Embodied R1.5-8B	RynnBrain1.1 9B	HY-Embodied 1.0-30B-A3B
Spatial Reasoning	CV-Bench	88.56	**89.99**	88.62	87.55	87.03	88.76	89.39
	VSI-Bench	60.83	70.69	75.30	51.58	64.78	**75.73**	59.47
	OmniSpatial	**58.84**	56.36	51.27	43.84	47.36	48.08	56.56
	MindCube	58.67	69.90	61.62	35.71	32.76	**79.24**	64.67
	RoboSpatial-Home	63.46	**72.80**	63.95	62.19	63.75	59.67	65.98
	All-Angles-Bench	**65.43**	64.12	50.70	53.85	50.19	56.19	62.85
	EmbSpatial-Bench	82.94	**84.59**	80.99	77.64	76.62	82.50	82.25
	ERQA	**60.00**	58.25	44.00	43.25	43.00	47.50	57.25
	OpenEQA	67.48	**69.51**	56.17	57.24	55.58	58.45	58.48
Action Guidance	RoboRefIt	84.45	**85.96**	83.16	77.63	83.48	82.09	82.20
	PointBench	76.66	**81.22**	53.91	58.01	61.72	53.63	69.33
	VABench-point	54.71	59.33	4.67	48.17	**72.34**	27.56	60.59
	PixMoPointsEval†	62.42	**74.06**	63.29	51.91	65.96	61.25	57.94
	where2place	57.61	73.52	66.51	57.86	73.15	**74.63**	60.74
	ShareRobot-Affor.	29.22	**30.84**	27.58	26.26	19.39	22.61	26.19
	PIO	64.40	**71.60**	58.13	54.43	62.80	64.33	62.17
	NaviTrace‡	30.87	**42.80**	-25.27	-11.18	-25.77	-12.39	-36.42
	VABench-trace§ ↓	139.28	108.15	173.44	173.29	**85.57**	177.42	132.40
	ShareRobot-Traj.§ ↓	0.3518	0.2390	0.3061	0.4169	0.3052	0.2757	**0.2317**
Temporal Under-standing	EgoTempo	38.20	**44.60**	30.00	22.40	32.60	32.40	19.20
	Video-MME	71.33	**74.19**	66.67	65.78	65.44	68.30	62.81
	MVBench	68.55	**71.32**	68.53	58.13	61.87	67.89	63.29
	LongVideoBench	60.48	**64.89**	64.14	57.49	59.48	61.23	59.40
	QVHighlights-TimeLens	54.82	**58.26**	39.37	4.55	47.53	31.22	35.07
State Verification	StateBench-P	65.80	**76.80**	73.40	61.40	60.60	61.20	55.80
	StateBench-T	43.76	**46.21**	33.39	34.39	32.06	37.52	37.38
General	MMMU	74.86	**76.19**	57.62	63.05	54.95	58.86	69.14
	MMVet	**80.09**	79.45	58.07	72.11	65.28	61.42	69.40
	RealWorldQA	82.09	**83.40**	73.46	68.50	69.80	76.73	76.99
	MMBench-EN	92.35	**92.40**	90.30	89.03	88.89	91.15	92.26
	IFEval	**92.42**	91.31	69.87	76.52	75.42	87.43	80.04
	MMLU-Pro	84.39	**84.77**	60.06	58.36	59.98	79.23	77.19
	BFCL-v3	58.75	**60.12**	10.00	6.88	23.75	46.88	-
	LiveCodeBench v6	**77.31**	72.91	30.87	37.00	28.63	47.80	44.27

All results are reproduced under the DeepInsight evaluation protocol; the BFCL entry for Hy-Embodied-VLM-1.0 is left blank because its native tool-calling format is not currently supported. ShareRobot-Affor. and ShareRobot-Traj. denote ShareRobot-Affordance and ShareRobot-Trajectory, respectively. Unless specified, all entries report the native benchmark metric (higher is better, in %). Exceptions include: † point-level F1 (%, higher is better); ‡ NaviTrace official semantic-aware normalized score (higher is better and may be negative); and § distance-based metrics, including VABench-trace trajectory RMSE (pixels) and ShareRobot-Trajectory discrete Fréchet distance (DFD), where lower is better. Bold and underlined values denote the best and second-best results. The blue column highlights Capek 0.5.

On the primary 35B-A3B track, Capek 0.5 improves 28 of 34 matched rows over the Qwen3.6 initialization. The gains concentrate on the execution-facing capabilities targeted by Capek 0.5. In Action Guidance, all 10 rows improve, covering referring and point grounding (PixMoPointsEval, $62.42 \rightarrow 74.06$), affordance placement (Where2Place, $57.61 \rightarrow 73.52$), ordered navigation traces (NaviTrace, $30.87 \rightarrow 42.80$), and trajectory error metrics (VABench-trace RMSE decreases by 31.13 pixels; ShareRobot-Trajectory DFD decreases by 0.1128). These results show that the Guidance specialist improves not only recognition-style grounding, but also action-oriented targets and paths that can be consumed by downstream embodied systems.

The other capability families support the same execution-centric view. Temporal Understanding improves on all five evaluated rows, indicating stronger event and long-video reasoning. Both State Verification rows improve as well, with Capek-StateBench-P increasing by 11.00 points and Capek-StateBench-T by 2.45 points, showing that the model better evaluates physical predicates and task progress after post-training. Spatial Reasoning improves on 6 of 9 rows, including embodied and relation-heavy benchmarks such as VSI-Bench, MindCube, RoboSpatial-Home, EmbSpatial-Bench, and OpenEQA. General retention remains largely preserved, with 5 of 8 controls improving. Taken together, the 35B-A3B results indicate that Capek 0.5 strengthens the perception–reasoning–guidance–verification loop central to embodied execution while maintaining broad general competence.

Overall, the results show that execution-centric post-training across the key stages of embodied execution yields a unified embodied model with stronger capabilities, while largely preserving general multimodal and language abilities.

Table 2: Capability specialization and consolidation on capability-specific 35B-A3B benchmark suites. All model variants are initialized from the same Qwen3.6-35B-A3B checkpoint and evaluated under a unified protocol. Mix-RL denotes the data-mixing reinforcement learning baseline.

		Specialization		Consolidation			
Capability	**Signature benchmark**	**Shared start**	**Matching expert**	**Mix-RL**	**TIES**	**MOPD**	**TIES +MOPD**
Spatial	MindCube	58.67	**71.52**	60.38	58.38	68.48	<u>69.90</u>
	VSI-Bench	60.83	<u>70.99</u>	65.05	**71.02**	69.54	70.69
Temporal	EgoTempo	38.20	**50.80**	42.80	<u>49.40</u>	43.20	44.60
	LongVideoBench	60.48	**65.31**	63.31	<u>65.14</u>	64.48	64.89
Guidance	VABench-point	54.71	59.00	58.33	57.67	**59.78**	<u>59.33</u>
	VABench-trace ↓	139.28	**100.43**	121.8	117.49	108.83	<u>108.15</u>
State	StateBench-P	65.80	**80.00**	70.40	68.00	75.80	<u>76.80</u>
	StateBench-T	43.76	<u>47.88</u>	**48.23**	43.55	45.52	46.21

4.2 Consolidation Analysis

Table 2 presents a controlled consolidation ablation on the 35B-A3B track, where each capability domain is analyzed using representative signature benchmarks. The shared start gives the pre-specialization reference, the matching expert provides the capability-acquisition reference for each row, and Mix-RL, TIES, MOPD, and TIES+MOPD compare alternative routes for consolidating the specialists into a single model.

First, domain-specific post-training produces substantial and consistent specialization gains. Starting from the same shared backbone, the matching experts improve all eight signature benchmarks, defining the capability gains that the consolidation stage aims to preserve. Second, the individual consolidation routes exhibit different capability biases: Mix-RL improves over the shared start across all benchmarks but achieves the best result only on StateBench-T; TIES preserves spatial and temporal capabilities particularly well, but introduces regressions on MindCube and StateBench-T relative to the shared start. MOPD provides a more uniform profile and performs especially well on action- and state-oriented tasks. Third, applying MOPD from the TIES-merged initialization yields the strongest overall consolidation trade-off. Relative to MOPD alone, it improves seven of the eight results while giving up only (0.45) points on VABench-point.

In conclusion, these results show that starting from the TIES-merged model allows subsequent MOPD training to retain consistently strong performance across all four capability domains while mitigating the capability trade-offs observed with either consolidation route alone.

Table 3: EmbodiedBench evaluation on EB-HAB and EB-ALF. All values are task success rates (%). Bold and underlined values denote the best and second-best results, respectively.

	EB-HAB							EB-ALF						
Model	**Avg.**	**Base**	**Comm.**	**Comp.**	**Spatial**	**Visual**	**Long**	**Avg.**	**Base**	**Comm.**	**Comp.**	**Spatial**	**Visual**	**Long**
Embodied-R1.5	21.7	56.0	6.0	18.0	22.0	12.0	16.0	14.7	20.0	18.0	14.0	18.0	12.0	6.0
RynnBrain1.1-9B	44.7	82.0	30.0	52.0	<u>34.0</u>	<u>48.0</u>	22.0	35.3	46.0	42.0	38.0	28.0	20.0	38.0
RoboBrain2.0-32B	38.3	68.0	22.0	<u>54.0</u>	22.0	42.0	22.0	25.0	34.0	24.0	34.0	20.0	26.0	12.0
RynnBrain-30B-A3B	26.7	66.0	14.0	34.0	22.0	22.0	2.0	26.3	32.0	30.0	34.0	22.0	26.0	14.0
Qwen3.6-35B-A3B	<u>46.0</u>	<u>86.0</u>	<u>32.0</u>	<u>54.0</u>	32.0	44.0	<u>28.0</u>	<u>50.7</u>	**52.0**	**60.0**	<u>48.0</u>	**44.0**	<u>44.0</u>	<u>56.0</u>
Capek 0.5-35B-A3B	**63.0**	**96.0**	**48.0**	**74.0**	**46.0**	**60.0**	**54.0**	**55.3**	**52.0**	<u>58.0</u>	**70.0**	<u>42.0</u>	**48.0**	**62.0**

Comm.: Common Sense; Comp.: Complex Instruction; Spatial: Spatial Awareness; Visual: Visual Appearance; Long: Long Horizon. Each subset contains 50 episodes. RynnBrain 1.1, Qwen 3.6, and Capek 0.5 are evaluated with thinking mode enabled. Embodied-R1.5, RoboBrain 2.0, and RynnBrain are evaluated using their respective native inference configurations.

4.3 Embodied Agent Evaluation

Beyond offline benchmarks, we evaluate whether the capabilities learned by Capek 0.5 transfer to embodied task execution. We therefore evaluate the 35B-A3B model variant in two complementary simulated environments: EmbodiedBench (Yang et al., 2025b), which measures end-to-end embodied task success, and VIGIL (Chen et al., 2026), which provides fine-grained diagnosis of both isolated capabilities and their composition during interaction. Both benchmarks are strictly held out for evaluation without any task-specific model training.

Table 4: VIGIL evaluation. Overall columns report primary score (Score), world completion (W), and benchmark success (B); the eight task-family columns report W/B. All values are percentages.

	Overall			Diagnostic Probes W/B				Compositional Tasks W/B			
Model	**Score**	W	B	**PG**	**DA**	**VS**	**SV**	**AI**	**SI**	**SM**	**CR**
Embodied-R1.5	31.7	33.4	31.7	74.4/**74.4**	**59.2/58.4**	10.4/10.4	**96.0**/83.2	14.4/14.4	0.8/0.8	6.4/6.4	5.6/5.6
Qwen3.6-35B-A3B	30.2	37.4	28.8	76.0/52.8	22.4/20.8	36.0/35.2	**96.0/85.6**	41.6/18.4	**11.2/7.2**	4.0/3.2	**12.0/7.2**
RynnBrain1.1-9B	29.1	34.3	23.4	73.6/48.0	36.0/26.4	47.2/16.8	**96.0**/84.0	16.8/9.6	0.0/0.0	1.6/0.0	3.2/2.4
RoboBrain2.0-32B	**38.1**	36.2	**34.3**	71.2/69.6	32.8/32.8	**71.2/71.2**	90.4/76.8	12.0/12.0	2.4/2.4	**8.0/8.0**	1.6/1.6
RynnBrain-30B-A3B	17.2	22.2	13.5	6.4/5.6	53.6/41.6	19.2/10.4	92.0/46.4	3.2/1.6	0.0/0.0	3.2/2.4	0.0/0.0
Capek 0.5-35B-A3B	36.4	**38.6**	32.2	**85.6**/72.8	19.2/19.2	36.8/35.2	**96.0**/84.8	**47.2/28.8**	9.6/4.0	6.4/5.6	8.0/**7.2**

Each family contains 125 episodes under the same episode pack, prompt, parser, observation-history policy, decoding configuration, and action budget. PG: pixel grounding; DA: distance approach; VS: view search; SV: state verification; AI: approach-and-interact; SI: search-and-interact; SM: sequential manipulation; CR: constraint resolving.

EmbodiedBench. We evaluate Capek 0.5 on the two high-level suites of EmbodiedBench, EB-Habitat (EB-HAB) and EB-ALFRED (EB-ALF), each comprising 300 multi-step embodied episodes. Performance is measured by task success rate (SR). All models are evaluated using the same episode sets and default benchmark settings.

As shown in Table 3, Capek 0.5 achieves the highest average success rate on both EB-HAB and EB-ALF, outperforming the scale-matched Qwen3.6 backbone by 17.0 and 4.6 percentage points, respectively. The gains are especially large on long-horizon tasks, where success increases by 26.0 points on EB-HAB and 6.0 points on EB-ALF. Capek 0.5 also improves the complex-instruction subsets in both environments (+20.0 on EB-HAB and +22.0 on EB-ALF). These gains indicate that consolidating complementary reasoning capabilities improves the maintenance of task context and the coordination of perception, planning, and interaction over extended execution sequences.

VIGIL. We further evaluate Capek 0.5 on VIGIL, a frozen embodied-agent benchmark comprising 1,000 balanced episodes in AI2-THOR (Kolve et al., 2017) and ProcTHOR (Deitke et al., 2022). VIGIL distinguishes successful interaction with the environment from correct terminal reporting by separately evaluating world completion and benchmark success. Following the original protocol, we report overall world completion (W), overall benchmark success (B), and benchmark success for each task family. Here, W measures whether the target world predicate is satisfied, whereas B additionally requires a correct terminal report. The four diagnostic families evaluate grounding, approach, search, and state verification in isolation, while the four compositional families assess how these capabilities combine through interaction, multi-step manipulation, and constraint resolution.

As shown in Table 4, Capek 0.5 improves both world completion and benchmark success over the Qwen3.6 backbone, increasing overall W from 37.4% to 38.6% and overall B from 28.8% to 32.2%. The largest gain appears in pixel grounding, indicating *more accurate localization* of task-relevant visual targets. The gains extend to compositional tasks: benchmark success rises from 18.4% to 28.8% on approach-and-interact and from 3.2% to 5.6% on sequential manipulation. Since these tasks require the agent to identify relevant targets, approach them, execute actions, and verify the state across multiple steps, the results suggest that execution-centric capability consolidation transfers beyond isolated reasoning benchmarks to embodied task execution.

Overall, the gains on both EmbodiedBench and VIGIL indicate that the capabilities learned by Capek 0.5 compose effectively during embodied interaction, leading to more robust task-level execution.

5 Conclusion

We presented Capek 0.5, a unified VLM for execution-centric embodied reasoning. Its post-training recipe derives Spatial Reasoning, Temporal Understanding, Action Guidance, and State Verification specialists followed by a consolidation stage. We evaluated this design at three complementary levels: public benchmark comparisons for the 2B and 35B-A3B tracks, a controlled study of capability retention from specialist experts to the unified model, and closed-loop evaluation in simulated embodied environments. Together, these evaluations connect broad embodied capability, specialist-to-unified retention, and capability composition during interactive task execution.

AI USE STATEMENT

We used large models to assist with part of training-data construction, as described in Section 2 and Appendix B. To ensure data quality, we applied automatic checks for media integrity, output-format validity, and annotation consistency, together with task-specific checks for visual support, coordinate and timestamp validity, and consistency between supervision targets and the underlying execution records. We filtered invalid or inconsistent samples and removed duplicates. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To support reproducibility, we describe our capability-aligned data construction in Section 2 and provide detailed data sources, preprocessing procedures, and quality-control steps in Appendix B. Training configurations and implementation details are documented in Appendix D, while evaluation protocols and scoring procedures are provided in Appendix C.

REFERENCES

Abbas Abdolmaleki, Saminda Abeyruwan, Joshua Ainslie, et al. Gemini robotics 1.5: Pushing the frontier of generalist robots with advanced embodied reasoning, thinking, and motion transfer. *arXiv preprint arXiv:2510.03342*, 2025.

ACE-Brain Team, Ziyang Gong, Haoming Gu, Zehang Luo, et al. Ace-brain-0.5: A unified embodied foundational model for physical agentic ai. *arXiv preprint arXiv:2607.04426*, 2026.

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, et al. On-policy distillation of language models: Learning from self-generated mistakes. *arXiv preprint arXiv:2306.13649*, 2023.

AgiBot-World Contributors, Qingwen Bu, Jisong Cai, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.

Michael Ahn, Anthony Brohan, Noah Brown, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Johan Bjorck, Zhiqi Li, Yunze Man, et al. Vesta: A generalist embodied reasoning model. *arXiv preprint arXiv:2606.20905*, 2026.

Zhongang Cai, Ruisi Wang, Chenyang Gu, Fanyi Pu, Junxiang Xu, Yubo Wang, Wanqi Yin, Zhitao Yang, Chen Wei, Tongxi Zhou, et al. Scaling spatial intelligence with multimodal foundation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7879–7890, 2026.

Kaiyuan Chen, Shuangyu Xie, Zehan Ma, Pannag R. Sanketi, and Ken Goldberg. Robo2vlm: Visual question answering from large-scale in-the-wild robot manipulation datasets. *arXiv preprint arXiv:2505.15517*, 2025a.

Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic. *arXiv preprint arXiv:2306.15195*, 2023.

Lin Chen, Xilin Wei, Jinsong Li, et al. Sharegpt4video: Improving video understanding and generation with better captions. *arXiv preprint arXiv:2406.04325*, 2024.

Ying Chen, Lihuang Fang, Rui Jiang, Mingxu Wang, Zhifeng Gu, Lei Yi, and Jie Chen. Done, but not sure: Disentangling world completion from self-termination in embodied agents. *arXiv preprint arXiv:2605.08747*, 2026.

Yukang Chen, Wei Huang, Baifeng Shi, Qinghao Hu, Hanrong Ye, Ligeng Zhu, Zhijian Liu, Pavlo Molchanov, Jan Kautz, Xiaojuan Qi, Sifei Liu, Hongxu Yin, Yao Lu, and Song Han. Scaling rl to long videos. 2025b.

Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Ronghao Dang, Jiayan Guo, Bohan Hou, et al. Rynnbrain: Open embodied foundation models. *arXiv preprint arXiv:2602.14979*, 2026.

Matt Deitke, Eli VanderBilt, Alvaro Herrasti, et al. Procthor: Large-scale embodied ai using procedural generation. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Matt Deitke, Christopher Clark, Sangho Lee, et al. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. *arXiv preprint arXiv:2409.17146*, 2024.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

Mengfei Du, Binhao Wu, Zejun Li, Xuanjing Huang, and Zhongyu Wei. Embspatial-bench: Benchmarking spatial understanding for embodied tasks with large vision-language models. *arXiv preprint arXiv:2406.05756*, 2024.

Thomas Eiter, Heikki Mannila, et al. Computing discrete fréchet distance. 1994.

Kaituo Feng, Manyuan Zhang, Hongyu Li, et al. Onethinker: All-in-one reasoning model for image and video. *arXiv preprint arXiv:2512.03043*, 2025.

Foundation Model Team, Li Auto Inc. Mach-mind-4-flash technical report. *arXiv preprint arXiv:2607.09375*, 2026.

Chaoyou Fu, Yuhan Dai, Yansong Luo, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. *arXiv preprint arXiv:2405.21075*, 2024.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 24108–24118, 2025.

Gemini Robotics Team. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Songhao Han, Wei Huang, Hairong Shi, Le Zhuo, Xiu Su, Shifeng Zhang, Xu Zhou, Xiaojuan Qi, Yue Liao, and Si Liu. Videoespresso: A large-scale chain-of-thought dataset for fine-grained video reasoning via core frame selection. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 26181–26191, 2025.

Xiaoshuai Hao, Yingbo Tang, Lingfeng Zhang, et al. Roboafford++: A generative ai-enhanced dataset for multimodal affordance learning in robotic manipulation and navigation. *arXiv preprint arXiv:2511.12436*, 2025a.

Xiaoshuai Hao, Lei Zhou, Zhijian Huang, et al. Mimo-embodied: X-embodied foundation model technical report. *arXiv preprint arXiv:2511.16518*, 2025b.

Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. Localizing moments in video with natural language. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Chengkai Hou, Kun Wu, Jiaming Liu, et al. Robomind 2.0: A multimodal, bimanual mobile manipulation dataset for generalizable embodied intelligence. *arXiv preprint arXiv:2512.24653*, 2025.

Wenlong Huang, Fei Xia, Ted Xiao, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

Wenlong Huang, Chen Wang, Ruohan Zhang, et al. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, et al. Editing models with task arithmetic. *International Conference on Learning Representations (ICLR)*, 2023.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination-free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Yuheng Ji, Huajie Tan, Jiayu Shi, et al. Robobrain: A unified brain model for robotic manipulation from abstract to concrete. *arXiv preprint arXiv:2502.21257*, 2025.

Mengdi Jia, Zekun Qi, Shaochen Zhang, Wenyao Zhang, Xinqiang Yu, Jiawei He, He Wang, and Li Yi. Omnispatial: Towards comprehensive spatial reasoning benchmark for vision language models. *arXiv preprint arXiv:2506.03135*, 2025.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, et al. Segment anything. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Eric Kolve, Roozbeh Mottaghi, Winson Han, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Chengshu Li, Ruohan Zhang, Josiah Wong, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. *Conference on Robot Learning (CoRL)*, 2023a.

Hongyu Li, Jinyu Chen, Ziyu Wei, et al. Llava-st: A multimodal large language model for fine-grained spatial-temporal understanding. *arXiv preprint arXiv:2501.08282*, 2025.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International Conference on Machine Learning (ICML)*, 2023b.

Kunchang Li, Yali Wang, Yinan He, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. *arXiv preprint arXiv:2311.17005*, 2024.

Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, et al. Grounded language-image pre-training. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Siyi Li, Chunyu Sun, Jiahao Zhang, Yuchen Kang, Wuliang Wang, Yu Qiu, Rui Jiang, Haitao Cui, and Jie Chen. Deepinsight: A unified evaluation infrastructure across the physical ai stack. *arXiv preprint arXiv:2606.17574*, 2026.

Jacky Liang, Wenlong Huang, Fei Xia, et al. Code as policies: Language model programs for embodied control. *arXiv preprint arXiv:2209.07753*, 2022.

Disheng Liu, Tuo Liang, Zhe Hu, Jierui Peng, Yiren Lu, Yi Xu, Yun Fu, Yu Yin, et al. Spatial intelligence in vision-language models: A comprehensive survey. 2025.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023a.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023b.

Yuan Liu, Haodong Duan, Yuanhan Zhang, et al. Mmbench: Is your multi-modal model an all-around player? *arXiv preprint arXiv:2307.06281*, 2023c.

Ruijie Lu, Yiyang Ma, Xiaokang Chen, et al. Thinking with visual primitives. *Technical report*, 2026.

Yuhao Lu, Yixuan Fan, Beixing Deng, Fangfu Liu, Yali Li, and Shengjin Wang. VL-Grasp: a 6-Dof interactive grasp policy for language-oriented objects in cluttered indoor scenes. *arXiv preprint arXiv:2308.00640*, 2023. URL https://arxiv.org/abs/2308.00640. IROS 2023.

Wenhan Ma, Jianyu Wei, Liang Zhao, et al. Mopd: Multi-teacher on-policy distillation for capability integration in llm post-training. *arXiv preprint arXiv:2606.30406*, 2026.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

NVIDIA. Cosmos 3: Omnimodal world models for physical ai. *arXiv preprint arXiv:2606.02800*, 2026. URL https://research.nvidia.com/labs/cosmos-lab/cosmos3/technical-report.pdf.

NVIDIA, Niket Agarwal, Arslan Ali, Maciej Bala, et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025a.

NVIDIA, Alisson Azzolini, Junjie Bai, et al. Cosmos-reason1: From physical common sense to embodied reasoning. *arXiv preprint arXiv:2503.15558*, 2025b.

Andreea-Maria Oncescu, Joao F Henriques, Yang Liu, Andrew Zisserman, and Samuel Albanie. Queryd: A video dataset with high-quality text and audio narrations. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2265–2269. IEEE, 2021.

Baiyu Pan, Daqin Luo, Junpeng Yang, et al. Thinker: A vision-language foundation model for embodied intelligence. *arXiv preprint arXiv:2601.21199*, 2026.

Shishir G. Patil, Huanzhi Mao, Charlie Cheng-Jie Ji, Fanjia Yan, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025.

Viorica Patraucean, Lucas Smaira, Ankush Gupta, Adria Recasens, Larisa Markeeva, Dylan Banarse, Skanda Koppula, Mateusz Malinowski, Yi Yang, Carl Doersch, et al. Perception test: A diagnostic benchmark for multimodal video models. *Advances in Neural Information Processing Systems*, 36: 42748–42761, 2023.

Zhiliang Peng, Wenhui Wang, Li Dong, et al. Kosmos-2: Grounding multimodal large language models to the world. *arXiv preprint arXiv:2306.14824*, 2023.

Chiara Plizzari, Alessio Tonioni, Yongqin Xian, Achin Kulshrestha, and Federico Tombari. Omnia de egotempo: Benchmarking temporal understanding of multi-modal llms in egocentric videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025a.

Chiara Plizzari, Alessio Tonioni, Yongqin Xian, Achin Kulshrestha, and Federico Tombari. Omnia de egotempo: Benchmarking temporal understanding of multi-modal llms in egocentric videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 24129–24138, 2025b.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL `https://qwen.ai/blog?id=qwen3.5`.

Qwen Team. Qwen3.6-35B-A3B: Agentic coding power, now open to all, April 2026b. URL `https://qwen.ai/blog?id=qwen3.6-35b-a3b`.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, 2021.

Sahar Salimpour, Lei Fu, Kajetan Rachwał, Pascal Bertrand, Kevin O'Sullivan, Robert Jakob, Farhad Keramat, Leonardo Militano, Giovanni Toffetti, Harry Edelman, et al. Towards embodied agentic ai: Review and classification of llm-and vlm-driven robot autonomy and interaction. *arXiv preprint arXiv:2508.05294*, 2025.

Pierre Sermanet, Tianli Ding, Jeffrey Zhao, et al. Robovqa: Multimodal long-horizon reasoning for robotics. *arXiv preprint arXiv:2311.00899*, 2023.

Zhihong Shao, Peiyi Wang, Qihao Zhu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Yijun Shen, Delong Chen, Xianming Hu, Jiaming Mi, Hongbo Zhao, Kai Zhang, and Pascale Fung. Reward prediction with factorized world states. *arXiv preprint arXiv:2603.09400*, 2026.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Gunnar A. Sigurdsson, Gul Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016.

Ishika Singh, Valts Blukis, Arsalan Mousavian, et al. Progprompt: Generating situated robot task plans using large language models. *arXiv preprint arXiv:2209.11302*, 2022.

Chan Hee Song, Valts Blukis, Jonathan Tremblay, Stephen Tyree, Yu Su, and Stan Birchfield. Robospatial: Teaching spatial understanding to 2d and 3d vision-language models for robotics. *arXiv preprint arXiv:2411.16537*, 2024.

Xiaowen Sun, Matthias Kerzel, Mengdi Li, et al. Statevlm: A state-aware vision-language model for robotic affordance reasoning. *arXiv preprint arXiv:2605.03927*, 2026.

Huajie Tan, Enshen Zhou, Zhiyu Li, et al. Robobrain 2.5: Depth in sight, time in mind. *arXiv preprint arXiv:2601.14352*, 2026.

Tencent Robotics X and HY Vision Team. Hy-embodied-0.5: Embodied foundation models for real-world agents. *arXiv preprint arXiv:2604.07430*, 2026.

Shengbang Tong, Ellis L Brown II, Penghao Wu, Sanghyun Woo, Adithya Jairam Iyer, Sai Charitha Akula, Shusheng Yang, Jihan Yang, Manoj Middepogu, Ziteng Wang, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Haoqing Wang, Xiang Long, Ziheng Li, Yilong Xu, Tingguang Li, and Yehui Tang. To mix or to merge: Toward multi-domain reinforcement learning for large language models. *arXiv preprint arXiv:2602.12566*, 2026a.

Haoqing Wang, Xiang Long, Ziheng Li, Yilong Xu, Tingguang Li, and Yehui Tang. To mix or to merge: Toward multi-domain reinforcement learning for large language models. *arXiv preprint arXiv:2602.12566*, 2026b.

Peng Wang, Shuai Bai, Sinan Tan, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024a.

Qineng Wang, Baiqiao Yin, Pingyue Zhang, et al. Mindcube: Spatial mental modeling from limited views. *arXiv preprint arXiv:2506.21458*, 2025.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems*, 2024b.

Ziyi Wang, Xumin Yu, Yongming Rao, Yonggen Ling, Yunheng Li, et al. Hy-embodied-vlm-1.0: Efficient physical-world agents. *arXiv preprint arXiv:2607.12894*, 2026c.

Tim Windecker, Manthan Patel, Moritz Reuss, Richard Schwarzkopf, Cesar Cadena, Rudolf Lioutikov, Marco Hutter, and Jonas Frey. Navitrace: Evaluating embodied navigation of vision-language models. *arXiv preprint arXiv:2510.26909*, 2025.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, et al. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. *International Conference on Machine Learning (ICML)*, 2022.

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*, 2024a.

Haoning Wu, Dongxu Li, Bei Chen, et al. Longvideobench: A benchmark for long-context interleaved video-language understanding. *arXiv preprint arXiv:2407.15754*, 2024b.

xAI. Grok-1.5 vision preview and realworldqa. `https://x.ai/news/grok-1.5v`, 2024. Accessed 2026-07-30.

Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Haotian Xue, Yunhao Ge, Yu Zeng, Zhaoshuo Li, Ming-Yu Liu, Yongxin Chen, and Jiaojiao Fan. Point-it-out: Benchmarking embodied reasoning for vision language models in multi-stage visual grounding. *arXiv preprint arXiv:2509.25794*, 2025.

Prateek Yadav, Derek Tam, Leshem Choshen, et al. Ties-merging: Resolving interference when merging models. *arXiv preprint arXiv:2306.01708*, 2023.

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 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10632–10643, June 2025a.

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. *arXiv preprint arXiv:2502.09560*, 2025b.

Chun-Hsiao Yeh, Chenyu Wang, Shengbang Tong, Ta-Ying Cheng, Rouyu Wang, Tianzhe Chu, Yuexiang Zhai, Yubei Chen, Shenghua Gao, and Yi Ma. Seeing from another perspective: Evaluating multi-view understanding in mllms. *arXiv preprint arXiv:2504.15280*, 2025.

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.

Haoxuan You, Haotian Zhang, Zhe Gan, et al. Ferret: Refer and ground anything anywhere at any granularity. *arXiv preprint arXiv:2310.07704*, 2023.

Le Yu, Bowen Yu, Haiyang Yu, et al. Language models are super mario: Absorbing abilities from homologous models as a free lunch. *International Conference on Machine Learning (ICML)*, 2024.

Weihao Yu, Zhengyuan Yang, Linjie Li, et al. Mm-vet: Evaluating large multimodal models for integrated capabilities. *arXiv preprint arXiv:2308.02490*, 2023.

Wentao Yuan, Jiafei Duan, Valts Blukis, et al. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024.

Yifu Yuan, Haiqin Cui, Yibin Chen, Zibin Dong, Fei Ni, Longxin Kou, Jinyi Liu, Pengyi Li, Yan Zheng, and Jianye Hao. From seeing to doing: Bridging reasoning and decision for robotic manipulation. *arXiv preprint arXiv:2505.08548*, 2025a.

Yifu Yuan, Haiqin Cui, Yaoting Huang, et al. Embodied-r1: Reinforced embodied reasoning for general robotic manipulation. *arXiv preprint arXiv:2508.13998*, 2025b.

Yifu Yuan, Yaoting Huang, Xianze Yao, et al. Embodied-r1.5: Evolving physical intelligence via embodied foundation models. *arXiv preprint arXiv:2606.11324*, 2026.

Xiang Yue, Yuansheng Ni, Kai Zhang, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Abhay Zala, Jaemin Cho, Satwik Kottur, Xilun Chen, Barlas Oguz, Yashar Mehdad, and Mohit Bansal. Hierarchical video-moment retrieval and step-captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23056–23065, 2023.

Jiahui Zhang, Yurui Chen, Yanpeng Zhou, Yueming Xu, Ze Huang, Jilin Mei, Junhui Chen, Yujie Yuan, Xinyue Cai, Guowei Huang, Xingyue Quan, Hang Xu, and Li Zhang. From flatland to space: Teaching vision-language models to perceive and reason in 3d. *arXiv preprint arXiv:2503.22976*, 2025a.

Jun Zhang, Teng Wang, Yuying Ge, Yixiao Ge, Xinhao Li, Ying Shan, and Limin Wang. Timelens: Rethinking video temporal grounding with multimodal llms. *arXiv preprint arXiv:2512.14698*, 2025b.

Yuanhan Zhang, Jinming Wu, Wei Li, et al. Llava-video: Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*, 2024.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

APPENDIX CONTENTS

A Related Work **18**
A.1 Robot-Centric VLMs and Embodied Systems 18
A.2 Grounded Visual Reasoning for Robot Scenes 18
A.3 Expert Specialization and Capability Consolidation 18

B Data Construction Details **19**
B.1 Spatial Reasoning 19
B.2 Temporal Understanding 20
B.3 Action Guidance 20
B.4 Data Standardization. 20

C Evaluation Protocol **21**
C.1 Evaluation Platform and Baselines 21
C.2 Capek-StateBench: A New Benchmark for State Verification 21
C.3 Benchmark Overview 22

D Training and Inference Details **22**
D.1 Training Details 23
D.2 Inference Settings 24
D.3 Evaluation Prompts 24
D.4 Capek-StateBench Protocol 26
D.5 Trajectory Reward and VABench-Trace Score 26

E Additonal Experimental Results **27**

F Qualitative EmbodiedBench Rollout **28**

G Qualitative VIGIL Rollouts **29**

H Qualitative Static QA Rollouts **30**
H.1 Static QA Failure Cases 34

A Related Work

A.1 Robot-Centric VLMs and Embodied Systems

Foundation models support embodied decision-making through diverse interfaces. SayCan grounds language-model proposals with learned affordances, Inner Monologue incorporates environmental feedback into reasoning, and Code as Policies and ProgPrompt represent plans as executable programs (Ahn et al., 2022; Huang et al., 2022; Liang et al., 2022; Singh et al., 2022). VoxPoser translates language instructions into composable 3D value fields, while PaLM-E integrates continuous sensor observations into a multimodal model (Huang et al., 2023; Driess et al., 2023). These approaches demonstrate how pretrained semantic knowledge can support planning and interaction.

Robot-centric VLMs further develop the perceptual and reasoning capabilities needed for execution. RoboPoint focuses on task-conditioned affordance pointing; RoboBrain and RoboBrain 2.5 address manipulation reasoning, trajectory prediction, depth estimation, and temporal understanding; and RynnBrain integrates visual grounding with embodied reasoning (Yuan et al., 2024; Ji et al., 2025; Tan et al., 2026; Dang et al., 2026). MiMo-Embodied, Thinker, HY-Embodied, Embodied-R1.5, ACE-Brain, Vesta, and Mach-Mind explore complementary directions in multi-task training, reinforcement learning, specialization, and capability integration (Hao et al., 2025b; Pan et al., 2026; Tencent Robotics X & HY Vision Team, 2026; Yuan et al., 2026; ACE-Brain Team et al., 2026; Bjorck et al., 2026; Foundation Model Team, Li Auto Inc., 2026). Related efforts extend foundation models toward action generation, as in Gemini Robotics, and physical-world prediction, as in Cosmos (Gemini Robotics Team, 2025; Abdolmaleki et al., 2025; NVIDIA et al., 2025a; NVIDIA, 2026).

Within this landscape, Capek 0.5 organizes embodied reasoning around four recurring execution needs and separates capability acquisition from consolidation. It retains a single autoregressive VLM interface, producing textual judgments and structured visual guidance for downstream systems.

A.2 Grounded Visual Reasoning for Robot Scenes

Grounded vision-language models connect language to boxes, masks, regions, and points (Li et al., 2022; Liu et al., 2023b; Kirillov et al., 2023; Peng et al., 2023; Chen et al., 2023; You et al., 2023). Molmo and PixMo scale point-based supervision, while Thinking with Visual Primitives and RynnBrain incorporate explicit visual references into reasoning (Deitke et al., 2024; Lu et al., 2026; Dang et al., 2026). These representations provide a foundation for the structured visual outputs used in Capek 0.5.

Beyond localization, robot-scene reasoning encompasses spatial relations and viewpoint transformations (Wang et al., 2025; Du et al., 2024), task-conditioned affordances and trajectories (Yuan et al., 2024; Windecker et al., 2025), and temporal reasoning across video frames (Xiao et al., 2021; Fu et al., 2024; Plizzari et al., 2025a). Embodied question answering further situates visual reasoning in navigation and manipulation contexts (Majumdar et al., 2024; Sermanet et al., 2023). These tasks involve complementary capabilities with distinct output formats and evaluation criteria.

State reasoning extends this scope to execution outcomes and progress. BEHAVIOR-1K provides structured household activities and physical predicates, StateVLM studies state-aware affordances, and factorized world-state prediction estimates progress from action–observation sequences (Li et al., 2023a; Sun et al., 2026; Shen et al., 2026). Building on these directions, Capek 0.5 unifies spatial, temporal, action-guidance, and state-verification capabilities through specialist training and consolidation while preserving their native supervision and evaluation semantics.

A.3 Expert Specialization and Capability Consolidation

Generalist mixtures and expert-first systems provide complementary routes to multi-capability models. Vesta studies a curated supervised mixture for embodied reasoning, MiMo-Embodied examines cross-domain embodied training, and ACE-Brain and Mach-Mind place specialization and reconciliation inside broader post-training systems (Bjorck et al., 2026; Hao et al., 2025b; ACE-Brain Team et al., 2026; Foundation Model Team, Li Auto Inc., 2026). M2RL directly analyzes how mixing and merging affect reinforcement-learning domains (Wang et al., 2026a). Together, these works motivate

an evaluation view that retains specialist references: a final checkpoint alone cannot show which skills were produced or retained during unification.

Parameter-space methods such as Model Soups, Task Arithmetic, TIES, and DARE combine same-origin checkpoints without adding inference-time experts (Wortsman et al., 2022; Ilharco et al., 2023; Yadav et al., 2023; Yu et al., 2024). Knowledge distillation and generalized on-policy distillation transfer teacher behavior to a student on teacher- or student-generated sequences (Hinton et al., 2015; Agarwal et al., 2023). MOPD extends the latter formulation by routing student rollouts to multiple domain teachers during integration (Ma et al., 2026).

Capek 0.5 follows an expert-first design: within each model track, Spatial, Temporal, Guidance, and State specialists are derived from the same shared checkpoint and consolidated through routed policy-space consolidation into one inference-time model. The report compares the unified checkpoint with all four fixed specialist references and a weight-space merge baseline, making retention across heterogeneous robot-centric capabilities the central empirical question.

B Data Construction Details

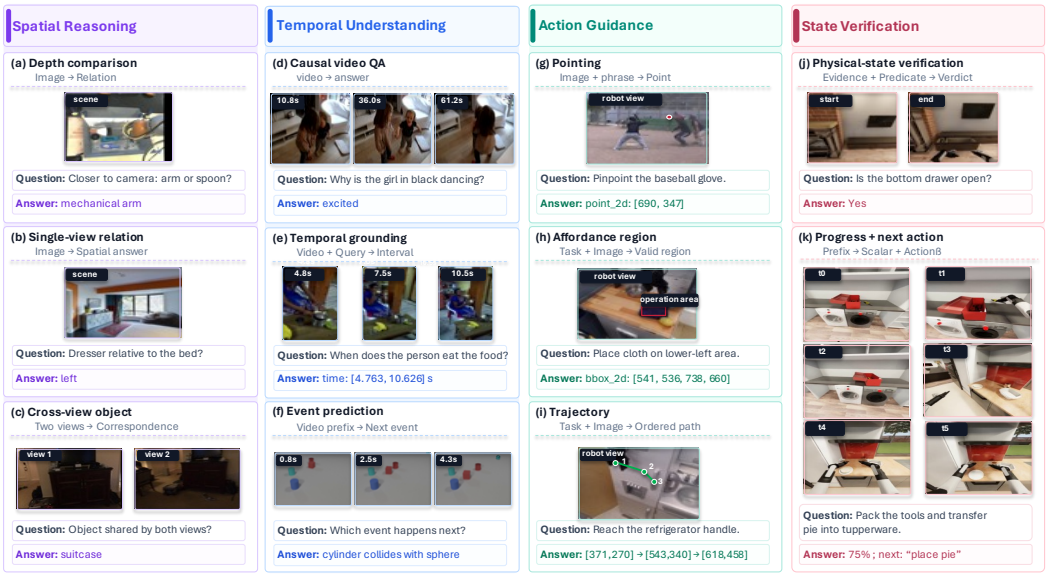


Figure 6: **Representative capability-aligned supervision.** Each panel shows a source-traceable input paired with its reference target, rather than a model prediction. The examples cover spatial relation and metric reasoning, temporal reasoning and localization, action-oriented points, regions, and trajectories, as well as physical-state, task-state, and progress records. These examples are grouped into four specialist branches according to their primary execution roles; multiple output formats within one family do not define separate specialists.

We show representative training samples of each capability in Figure 6.

B.1 Spatial Reasoning

We construct the Spatial post-training corpus from SenseNova-SI-8M (Cai et al., 2026), VSI-590K (Yang et al., 2025a), MindCube (Wang et al., 2025), and EmbSpatial-SFT (Du et al., 2024). These sources provide complementary annotations at the scene and viewpoint levels. The resulting dataset spans four categories: (i) *spatial relations*, including relative direction, orientation, and cardinal-relation reasoning; (ii) *metric reasoning*, including depth, distance, and size estimation; (iii) *multi-view reasoning*, including object correspondence and viewpoint transformation; and (iv) *spatial scene reasoning*, including object localization, counting, and navigation-relevant questions whose targets remain spatial relations or scene-level judgments rather than executable action paths. Before post-training, we filter records for annotation validity, visual support, target consistency, and

duplication, and then convert the retained samples into format-specific supervision. Difficulty-aware rollout filtering is applied later during specialist RL rather than treated as a static property of the corpus.

B.2 Temporal Understanding

The temporal data covers three complementary forms of supervision. (i) *Closed-form video reasoning* develops temporal reasoning through questions about actions, object interactions, event ordering, counting, and causal relationships. We curate these samples from CLEVRER (Yi et al., 2019), NExT-QA (Xiao et al., 2021), PerceptionTest (Patraucean et al., 2023), STAR (Wu et al., 2024a), LLaVA-Video (Zhang et al., 2024), and LongVideoReason (Chen et al., 2025b), covering both short clips and long-form videos. (ii) *Temporal grounding* trains the model to localize when an event occurs by predicting an explicit temporal interval $[t_{\text{start}}, t_{\text{end}}]$. We collect grounding annotations from Charades (Sigurdsson et al., 2016), DiDeMo (Hendricks et al., 2017), HiREST (Zala et al., 2023), QuerYD (Oncescu et al., 2021), and LLaVA-ST (Li et al., 2025). Some samples additionally include spatial localization, enabling joint spatial-temporal grounding. For these records, temporal localization remains the primary target, while frame-level regions provide an auxiliary spatial component. (iii) *General video understanding* provides richer execution semantics in open-domain settings, including robot manipulation, egocentric interaction, and general video description. These samples are collected from ShareGPT4Video (Chen et al., 2024), VideoEspresso (Han et al., 2025), and Cosmos-Reason1 variants built on robot-video corpora (NVIDIA et al., 2025b). We remove samples with missing timestamps, inverted or out-of-range intervals, inconsistent event boundaries, or missing spatial annotations required for spatiotemporal reasoning.

B.3 Action Guidance

We construct the Guidance supervision from four complementary task groups. (i) *Visual grounding* predicts object regions referred to by natural-language expressions. (ii) *Pointing and affordance localization* identify task-relevant interaction points or regions, including explicit negative cases in which no valid target exists. (iii) *Trajectory prediction* produces ordered image-space waypoint sequences for manipulation or navigation. (iv) *Embodied instruction grounding* links task language to the relevant referent, action type, interaction region, or immediate visual target.

The data are curated from PixMo (Deitke et al., 2024), RoboPoint (Yuan et al., 2024), DROID (Khazatsky et al., 2024), AgiBot World (AgiBot-World Contributors et al., 2025), RoboMIND 2.0 (Hou et al., 2025), FSD (Yuan et al., 2025a), ShareRobot (Ji et al., 2025), RoboVQA (Sermanet et al., 2023), Robo2VLM (Chen et al., 2025a), RoboAfford++ (Hao et al., 2025a), SPAR (Zhang et al., 2025a), and OneThinker (Feng et al., 2025), covering manipulation, navigation, embodied question answering, and robot instruction following. To enable unified post-training, all coordinate-based targets are converted from their native annotations to a shared $[0, 1000]^2$ image-coordinate system. Point, box, and affordance annotations preserve their original target semantics, including explicit negative cases and multiple valid interaction locations where applicable. Trajectory annotations retain ordered waypoint sequences rather than unordered point sets; navigation trajectories additionally preserve sparse semantic waypoints and, when available, denser 2D paths. This representation preserves trajectory order, endpoints, and path geometry for downstream verification while standardizing the supervision format across guidance tasks. Representative examples are shown in Figure 6.

B.4 Data Standardization.

After capability-specific construction, all samples are converted into a unified record format while preserving their original supervision targets. Every sample undergoes automatic validation to ensure media integrity, annotation consistency, and format correctness. Depending on the task, additional semantic checks verify that supervision targets remain visually supported after preprocessing. Dataset splits are inherited from the original sources and assigned before view serialization so that all aligned views of the same logical sample remain in the same split. Duplicate removal is performed at the logical-sample level to avoid leakage across training and evaluation.

C Evaluation Protocol

C.1 Evaluation Platform and Baselines

We systematically evaluate Capek 0.5-35B-A3B and Capek 0.5-2B on a various of embodied tasks, following the capability groups defined above. All results are produced with DeepInsight (Li et al., 2026), our evaluation harness, which records the executed configuration, generations, and scores of every run. For each benchmark, the dataset version, media sampling, prompt template, answer parser, scorer, and judge (when applicable) are identical across all compared models. Decoding configurations follow a two-level policy: external baselines use the inference settings recommended by their official papers or model cards, whereas each matched Capek 0.5–Qwen pair additionally shares an identical decoding configuration, so that signed differences within a pair can be attributed to the post-training recipe.

We report the 35B-A3B track as the main comparison and the 2B track as a compact-capacity counterpart, using the same benchmark rows for both. We include the primary 35B-A3B results and ranks A3B MoE models together with 7–9B dense models whose active parameter counts are comparable, following the active-compute comparison principle used by HY-Embodied-VLM-1.0 (Wang et al., 2026c). Table 8 then reports the 2B track against models with at most 4B parameters. The external baselines shown in the tables are Qwen (Bai et al., 2025a), RynnBrain and RynnBrain1.1 (Dang et al., 2026), MiMo-Embodied (Hao et al., 2025b), Embodied-R1.5 (Yuan et al., 2026), RoboBrain-family checkpoints (Ji et al., 2025; Tan et al., 2026), and HY-Embodied-VLM-1.0 (Wang et al., 2026c). All completed entries in both tables, including those for external baselines, are DeepInsight reruns; no scores are quoted from official reports.

C.2 Capek-StateBench: A New Benchmark for State Verification

Existing public benchmarks provide broad coverage of spatial reasoning, temporal understanding, and action guidance, but they rarely evaluate whether a model can verify the resulting world state after execution or assess task progress toward completion. To evaluate this capability, we introduce **Capek-StateBench**, a benchmark for State Verification consisting of two complementary tracks: **Capek-StateBench-P** for physical-state verification and **Capek-StateBench-T** for task-state verification.

Physical-State Verification (Capek-StateBench-P). The physical track evaluates whether a model can determine observable object states and spatial relations at annotated execution boundaries. Each sample consists of a single image, a short video clip, or a small set of key frames, together with a verification query. Queries either require selecting the correct state from predefined alternatives or verifying a physical predicate using a binary `yes`/`no` response.

Task-State Verification (Capek-StateBench-T). The task track evaluates a model's understanding of execution progress from a causal visual prefix. Given a high-level task instruction and the observed execution history, the model predicts (1) the current task progress and (2) the immediate next action required to continue execution; this track therefore instantiates the execution-level progress-and-next-step readout used during State Verification training (Section 3). Progress is defined using two complementary annotation schemes: task-condition progress measures the fraction of satisfied BEHAVIOR-1K (Li et al., 2023a) goal conditions, while primitive-skill progress measures the completion ratio of annotated low-level skills, including navigation. The released 500-example split combines both schemes, with 213 task-condition records and 287 primitive-skill records. The primary score is computed over the combined split, while annotation-type breakdowns are retained as diagnostics; Appendix D.4 gives the complete protocol.

Evaluation Metrics. For the physical track, we report normalized exact-match accuracy together with breakdowns by query type, evidence source, state category, and execution level. For the task track, progress is scored using $s_p = \max\left(0, 1 - \frac{|\hat{p}-p|}{25}\right)$, while next-step prediction is evaluated by semantic exact matching, denoted by s_a. The primary metric combines both aspects: $s_{\mathrm{T}} = \frac{5s_p+4s_a}{9}$. We additionally report progress mean absolute error, progress score, next-step accuracy, terminal versus non-terminal performance, label-space breakdowns, and output validity as diagnostic metrics. Qualitative examples from both Capek-StateBench tracks, together with the corresponding model-visible evidence and complete model rollouts, are provided in Appendix H.

C.3 Benchmark Overview

We organize the offline evaluation according to the primary capability exercised by each benchmark. Each benchmark is assigned to a single capability group based on its dominant prediction type, while retaining its original evaluation protocol and native metric.

• **Spatial Reasoning.** Benchmarks cover geometric reasoning, spatial relations, viewpoint understanding, and embodied scene understanding, including CV-Bench (Tong et al., 2024), VSI-Bench (Yang et al., 2025a), OmniSpatial (Jia et al., 2025), MindCube (Wang et al., 2025), RoboSpatial-Home (Song et al., 2024), All-Angles-Bench (Yeh et al., 2025), EmbSpatial-Bench (Du et al., 2024), ERQA (Gemini Robotics Team, 2025), and OpenEQA (Majumdar et al., 2024).

• **Temporal Understanding.** Benchmarks cover event ordering, temporal localization, and long-video understanding, including EgoTempo (Plizzari et al., 2025b), Video-MME (Fu et al., 2025), MVBench (Li et al., 2024), LongVideoBench (Wu et al., 2024b), and QVHighlights-TimeLens (Zhang et al., 2025b). For evaluation, QVHighlights-TimeLens reports temporal mIoU.

• **Action Guidance** covers visual grounding, pointing, affordance localization, placement, and trajectory prediction, where instructions must be grounded into actionable image-space outputs, including RoboRefIt (Lu et al., 2023), PointBench (Abdolmaleki et al., 2025), VABench-point/trace (Yuan et al., 2025a), PixMoPointsEval (Deitke et al., 2024), Where2Place (Yuan et al., 2024), ShareRobot-Affordance/Trajectory (Ji et al., 2025), PIO (Xue et al., 2025), and NaviTrace (Windecker et al., 2025). For evaluation, box grounding rows use IoU@0.5 in the benchmark coordinate system. ShareRobot-Affordance reports mean IoU over predicted and reference affordance boxes, with missing or invalid predictions counted as zero. Point rows use their benchmark-native point or mask scorers, including PixMoPointsEval point-level F1 and the PIO point-in-mask score. Trajectory rows keep their native trajectory metrics: NaviTrace reports its semantic-aware normalized score, VABench-trace reports trajectory RMSE, and ShareRobot-Trajectory reports raw discrete Fréchet distance.

• **State Verification.** Capek-StateBench-P and Capek-StateBench-T correspond to the physical and task levels of State Verification. The Physical track tests local object states and relations and is scored by normalized exact-match accuracy. The Task track probes task-state understanding through execution-progress estimation and immediate next-step prediction; we report the deterministic progress score, next-step accuracy, and their predeclared joint score.

• **General retention.** General multimodal and language benchmarks serve as regression controls for measuring whether general capabilities are retained after embodied post-training, including MMMU (Yue et al., 2024), MMVet (Yu et al., 2023), RealWorldQA (xAI, 2024), MMBench-EN (Liu et al., 2023c), IFEval (Zhou et al., 2023), MMLU-Pro (Wang et al., 2024b), BFCL-v3 (Patil et al., 2025), and LiveCodeBench v6 (Jain et al., 2024). Together, they check retention of broad multimodal understanding, instruction following, reasoning, function-calling, and coding abilities.

D Training and Inference Details

D.0.1 Capability-Specific Reward Design

Table 5 lists the output formats and their scoring rules.

Spatial Reasoning. Spatial reasoning tasks span relational understanding, metric estimation, cross-view correspondence, and viewpoint reasoning. Their outputs primarily follow standard verifiable reasoning formats, including multiple-choice, Boolean, set-valued, and numerical answers. Accordingly, rewards are computed using exact or set agreement, numerical tolerance, or one-to-one correspondence matching according to the output type.

Temporal Understanding. The Temporal Understanding specialist covers video understanding, event ordering, temporal localization, and space–time grounding. Categorical reasoning tasks reuse the scoring rules from Spatial Reasoning. For temporal localization, a valid predicted interval $I_p = [s_p, e_p]$ is compared with the reference interval $I_g = [s_g, e_g]$ using temporal Intersection-over-Union (tIoU) (Hendricks et al., 2017):

$$R_{\mathrm{tIoU}}(I_p, I_g) = \frac{\max(0, \min(e_p, e_g) - \max(s_p, s_g))}{\max(e_p, e_g) - \min(s_p, s_g)}. \tag{4}$$

Table 5: **Reward formulation for different output formats.** Each output format combines binary format validity with one or more task-specific reward functions. The same scoring functions are reused whenever identical output types appear across specialists.

Reward term	Specialist(s)	Applied format(s)	Scoring rule
Format validity	All four	Every declared output format	Binary success of format-specific hard parsing
Answer correctness	Spatial Reasoning, Temporal Understanding	Choice, set, Boolean, scalar, OCR, and QA	Exact or set agreement, numerical tolerance, edit similarity, or frozen semantic score
Temporal overlap	Temporal Understanding	Event interval and space–time localization	Temporal IoU; space–time formats average temporal and frame-level box overlap
Box overlap	Temporal Understanding, Action Guidance	Frame-level space–time boxes and grounded box sets	Frame-averaged or one-to-one 2D IoU, with missing and unmatched items contributing zero
Point / region / affordance	Action Guidance	Pointing, region, affordance, and no-target formats	Point F1, target-mask hit, symmetric nearest-neighbor score, or exact abstention
Trajectory agreement	Action Guidance	Ordered waypoint sequence	Exponential reward over raw DFD after coordinate normalization and five-point arc-length resampling
Physical-state correctness	State Verification	Direct-state selection and binary predicate verification	Normalized exact agreement with the target state token or `yes`/`no` verdict
Progress and next-step agreement	State Verification	Progress-and-next-action JSON	Truncated progress proximity, normalized next-action agreement, and binary JSON validity

Predictions that fail format validation receive zero reward. For space–time grounding, temporal and spatial evidence are weighted equally: $R_{\mathrm{st}} = \frac{1}{2}R_{\mathrm{tIoU}} + \frac{1}{2}\overline{R}_{\mathrm{box}}$, where $\overline{R}_{\mathrm{box}}$ denotes the mean box IoU over the annotated frames, with missing boxes contributing zero.

Action Guidance. The Action Guidance specialist covers visual grounding, pointing, affordance localization, and trajectory prediction (Lu et al., 2026; Yuan et al., 2025b). Set-valued grounding tasks use one-to-one matching, with unmatched predictions or targets contributing zero. Region and affordance predictions are evaluated using their corresponding mask- or distance-based metrics, while explicit no-target cases reward correct abstention.

Ordered trajectories are evaluated using the discrete Fréchet distance (DFD), which preserves waypoint order (Eiter et al., 1994). Coordinates are first mapped to $[0,1]^2$, after which both predicted and reference paths are resampled at five uniformly spaced arc-length positions, yielding $\bar{P}$ and $\bar{Q}$. We use raw DFD as the geometric discrepancy and convert it to a higher-is-better reward with an exponential kernel:

$$C_{\mathrm{traj}}(\bar{P},\bar{Q}) = D_{\mathrm{DFD}}(\bar{P},\bar{Q}), \qquad R_{\mathrm{traj}}(\bar{P},\bar{Q}) = \exp\big(-\lambda_{\mathrm{traj}} C_{\mathrm{traj}}(\bar{P},\bar{Q})\big), \quad \lambda_{\mathrm{traj}} = 10. \tag{5}$$

For valid trajectories, the reward lies in $(0,1]$, with exact matches receiving one and larger geometric discrepancies receiving exponentially smaller rewards. Invalid or empty trajectories receive zero reward through format validation. Appendix D.5 provides the complete DFD definition and details.

State Verification. The State Verification specialist comprises two supervision types: Physical-State Verification (PSV) and Progress-Value Estimation (PVE). At the physical level, PSV predicts normalized object states or binary physical predicates from visual observations, and rewards exact agreement with the reference state or `yes`/`no` verdict. At the task level, PVE estimates execution progress from the observed execution history and, when available, predicts the annotated immediate next action. Progress supervision follows the goal-condition and child-skill decompositions described in Section 2.

Given predicted and reference progress values $\hat{p}, p \in [0,100]$, the progress reward is defined as a 25-percentage-point truncated proximity score: $r_{\mathrm{value}} = \max\left(0, 1 - \frac{|\hat{p}-p|}{25}\right)$. For samples with next-action supervision, the overall reward is $R_{\mathrm{prog+act}} = 0.5r_{\mathrm{value}} + 0.4r_{\mathrm{action}} + 0.1r_{\mathrm{fmt}}$, where r_{action} denotes normalized agreement with the annotated next action (including `null` for completed tasks), and r_{fmt} indicates JSON validity. Samples without next-action supervision are optimized using only r_{value} after the same format validation.

D.1 Training Details

We initialize Capek 0.5 for post-training from the Qwen vision-language model series, using Qwen3.6-35B-A3B and Qwen3.5-2B as the respective base models. Table 6 reports the available frozen 35B-A3B GRPO manifests. The 2B track uses the same verifier and optimization interfaces with branch-level settings recorded in its own run manifests. Training uses the verl (Sheng et al., 2024) and Megatron-LM (Shoeybi et al., 2019) stack with vLLM (Kwon et al., 2023) rollouts and bfloat16 precision.

Table 6: Recorded 35B-A3B specialist GRPO settings. Affordance and trajectory are two optimization contracts within the Guidance expert.

Setting	Spatial	Guidance (affordance)	Temporal	Guidance (trajectory)	State
Learning rate	1×10^{-6}	1×10^{-6}	1×10^{-6}	1×10^{-6}	1×10^{-6}
Rollouts per prompt	8	8	8	8	8
Actor KL coefficient	0.01	0.01	0.01	off	0.01
Global batch size	128	128	64	64	64
Maximum prompt length	8,192	5,120	10,240	10,240	10,240
Maximum response length	5,120	5,120	5,120	8,192	8,192
Train parallelism	TP2/EP8	TP2/EP8	TP2/EP8	TP2/EP8	TP2/EP8
Rollout tensor parallelism	8	4	4	4	4

The policy-space integration run initializes the student from the TIES-merged checkpoint and uses a global batch size of 240, learning rate 1×10^{-6}, two student rollouts per prompt, maximum prompt and response lengths of 10,240 and 5,120, and one training epoch. It uses 12 actor GPUs with TP2/EP4 and rollout TP2. The same four expert checkpoints used by TIES are served as frozen MOPD teachers. Each training contract is routed to its responsible expert; the run manifest fixes the contract-to-expert mapping, teacher identifiers, and route proportions.

D.2 Inference Settings

For Capek 0.5 and its shared Qwen start, the released DeepInsight configuration uses vLLM with a 128K model context, automatic truncation, at most 64 video frames, temperature 0.7, top-p 0.95, top-k 20, up to 16,384 generated tokens, and thinking enabled. The postprocessor removes `<think>` and optional `<answer>` wrappers before benchmark scoring. Tasks then apply their declared native format: for example, grounding uses IoU matching at 0.5 in the shared $[0, 1000]$ coordinate system, ShareRobot-Affordance reports mean IoU, point tasks use benchmark-native point or mask matching, and trajectory tasks use the corresponding normalized score, RMSE, or raw DFD scorer. Model-judged rows share one non-thinking judge profile. DeepInsight records the executed configuration, generations, and scores for each run (Li et al., 2026). Appendix D.3 gives the prompts and output formats used for the different benchmark types.

D.3 Evaluation Prompts

All public benchmarks are run with DeepInsight (Li et al., 2026). For each benchmark, the media input, question, prompt, parser, and scorer are fixed across models. Capek 0.5 and the corresponding Qwen checkpoint use a 128K context window, at most 64 video frames, temperature 0.7, top-p 0.95, top-k 20, and at most 16,384 output tokens. Other model families use the inference settings recommended by their papers or model cards. Following the benchmark-by-benchmark presentation of RynnBrain (Dang et al., 2026), Table 7 lists the text templates used in our evaluation. Media is supplied before the text and is omitted from the templates below.

Table 7: Prompt templates used for public benchmark evaluation. Braced terms are fields supplied by the dataset.

Benchmark	Prompt template
General	`{question}`

LongVideoBench	`{question}`
	`{options}`
	`Answer with the option letter from the given choices directly.`

RoboRefIt	`Output the target location with the form of a bounding box.`
	`Output format: [{"bbox_2d": [x1, y1, x2, y2], "label": "label"}].`
	`If no target is found, respond with an empty list.`
	`Question: {question}`

Continued from the previous page

Benchmark	Prompt template

ShareRobot-Affordance	`{question}`
	`Return only the affordance bounding box:`
	`[{"bbox_2d": [x1, y1, x2, y2], "label": "affordance area"}]`

PixMoPointsEval	`Locate all instances of {question} in the image.`
	`Output point coordinates as a JSON list using 0-1000 coordinates.`
	`If no instance is found, respond Not found.`

PIO	`Given the image and language description, output one to three points that best localize the description.`
	`Output format: [{"point_2d": [x, y], "label": "point_1"}].`
	`Language description: {question}`

VABench-point	`{question}`
	`Answer with a list of 2D points in JSON format.`
	`Output format: [{"point_2d": [x, y], "label": "label"}]`

where2place	`{question}`
	`Return several valid points inside the requested vacant area as a JSON array.`
	`Coordinates are normalized between 0 and 1000.`
	`Return only the JSON array.`

PointBench	`{hint}`
	`Return the requested point coordinates as a JSON array.`
	`Coordinates are normalized between 0 and 1000.`
	`For counting questions, return one point for each target instance.`
	`Return only the JSON array.`

VABench-trace / ShareRobot-Trajectory	`{question}`
	`Answer with a list of 2D points in JSON format.`
	`Output format: [{"point_2d": [x, y], "label": "waypoint N"}, ...]`

NaviTrace	`System: Act as a navigation expert. Start near the bottom center, adapt the path to the specified embodiment, stop for unsafe traffic conditions, and output only 2D points in JSON format.`
	`Embodiment: {hint}`
	`Task: {question}`
	`Predict a feasible path forward from the current first-person view.`

RoboSpatial-Home	`{question}`
	`For pinpoint or localization questions, return normalized points as a JSON array.`
	`For yes/no questions, answer only Yes or No.`

RoboSpatial judge	`Question: {question}`
	`Reference answer: {gold}`
	`Model prediction: {pred}`
	`Return only Correct or Incorrect.`

Before scoring, we remove the `<think>` span and unwrap an optional `<answer>` block. Bounding-box grounding rows use IoU@0.5 in $[0, 1000]$ coordinates, while ShareRobot-Affordance reports mean IoU over predicted and reference affordance boxes with invalid or missing predictions assigned zero. Pointing rows retain their benchmark-native scorers: PixMoPointsEval uses assignment-based point-level F1, and PIO uses point-in-mask scoring. Trajectory rows are not collapsed to a single transform: NaviTrace keeps its official

semantic-aware normalized protocol (Windecker et al., 2025), VABench-trace reports trajectory RMSE, and ShareRobot-Trajectory reports raw discrete Fréchet distance.

D.4 CAPEK-STATEBENCH PROTOCOL

Capek-StateBench contains two 500-example tracks aligned with the State Verification training objectives. Capek-StateBench-P evaluates physical-state verification from local image or video evidence. Predictions are normalized to the state label licensed by the question and scored by exact match; invalid or ambiguous answers receive zero. The primary PSV score is the mean accuracy over the complete split.

Capek-StateBench-T evaluates progress-value estimation from a high-level task instruction and an image-based causal visual prefix ending at the current checkpoint; future observations are excluded. The split contains 213 task-condition records and 287 primitive-skill records. For task-condition records, progress is the percentage of satisfied BEHAVIOR-1K goal conditions, while for primitive-skill records it is the percentage of completed annotated child skills, including navigation:

$$p_t^{\text{cond}} = 100\frac{|\mathcal{G}_t^{\text{sat}}|}{|\mathcal{G}|}, \qquad p_t^{\text{skill}} = 100\frac{n_t}{N}. \tag{6}$$

The two definitions remain separately typed. The next-action target is the immediate annotated continuation of the reference execution, rather than a claim of an optimal policy action.

For a valid progress prediction $\hat{p}_i \in [0, 100]$, the progress score is

$$s_i^{\text{prog}} = \max\left(0,\ 1 - \frac{|\hat{p}_i - p_i|}{25}\right); \tag{7}$$

missing, malformed, or out-of-range values receive zero. Next actions are scored by a fixed semantic judge that accepts harmless paraphrases but requires the operation, object, target, direction, and relation to agree with the reference. Letting $s_i^{\text{act}} \in \{0, 1\}$ denote this result, the task-track score is

$$s_i^{\text{task}} = \frac{5s_i^{\text{prog}} + 4s_i^{\text{act}}}{9}. \tag{8}$$

The primary Capek-StateBench-T result is the mean over all 500 records; results by progress annotation type, scenario type, and terminal status are retained as diagnostics. Full prompts and parser implementations are provided in the released DeepInsight configurations.

D.5 TRAJECTORY REWARD AND VABENCH-TRACE SCORE

For nonempty predicted waypoints $P = (p_1, \ldots, p_n)$ and reference waypoints $Q = (q_1, \ldots, q_m)$, coordinates are first mapped to $[0, 1]^2$. Both paths are then resampled at uniform arc-length positions to a common length, producing $\bar{P}$ and $\bar{Q}$; a zero-length constant path is represented by repeating its single location. A coupling $\gamma = ((i_\ell, j_\ell))_{\ell=1}^{L}$ starts at $(1, 1)$, ends at the final index pair, and has increments in $\{(1, 0), (0, 1), (1, 1)\}$. Let Γ be the set of these endpoint-preserving monotone couplings. The discrete Fréchet distance is

$$D_{\text{DFD}}(\bar{P}, \bar{Q}) = \min_{\gamma \in \Gamma} \max_{(i,j) \in \gamma} \|\bar{p}_i - \bar{q}_j\|_2. \tag{9}$$

Equation 5 applies the frozen geometric kernel to this distance. Invalid or empty parsed trajectories receive zero. The constant-path convention above makes resampling total on every nonempty valid path. Arc-length resampling removes a spurious dependence on waypoint count while the discrete Fréchet distance preserves path order and worst-case alignment.

VABench-trace uses a separate evaluation path. Its recorded evaluator applies the benchmark coordinate normalization and path preprocessing to each episode, maps trajectories into the padded-square pixel coordinate system, and resamples a predicted path when its waypoint count differs from the reference. The public table reports the global trajectory RMSE over all valid waypoint pairs,

$$S_{\text{VABench-trace}} = \sqrt{\frac{1}{M}\sum_{j=1}^{M} \|\hat{q}_j - q_j\|_2^2}, \tag{10}$$

where M is the total number of compared waypoints after preprocessing. This value is lower-is-better and remains in the benchmark pixel coordinate scale. The training reward remains the resampled-path kernel in Equation 5; it is not used to transform the public VABench-trace table entries.

Table 8: Benchmark results for the 2B variant, compared with models of at most 4B parameters.

Task family	Benchmark	Qwen3.5 2B	**Capek0.5 2B**	RynnBrain1.1 2B	RoboBrain 2.5-4B
Spatial Reasoning	CV-Bench	83.10	86.02	86.49	**86.90**
	VSI-Bench	37.05	55.00	**73.83**	49.80
	OmniSpatial	46.18	**50.16**	39.73	41.62
	MindCube	36.00	47.62	**54.86**	30.48
	RoboSpatial-Home	25.52	38.13	**59.95**	39.71
	All-Angles-Bench	49.11	**53.38**	44.28	47.37
	EmbSpatial-Bench	76.13	**78.68**	72.80	73.30
	ERQA	35.00	**46.25**	41.25	43.75
	OpenEQA	51.48	**57.78**	50.70	56.07
Action Guidance	RoboRefIt	76.50	**79.89**	78.97	3.01
	PointBench	56.83	**68.55**	51.76	67.13
	VABench-point	7.13	**36.00**	14.44	25.27
	PixMoPointsEval†	33.90	52.94	52.01	**61.02**
	where2place	29.16	35.86	66.46	**70.42**
	ShareRobot-Affordance	18.33	29.18	24.33	**50.43**
	PIO	25.17	49.76	**59.60**	58.43
	NaviTrace‡	-42.08	**12.59**	-20.74	-20.43
	VABench-trace§ ↓	255.47	160.63	216.83	**155.10**
	ShareRobot-Trajectory§ ↓	0.5689	0.2683	0.2823	**0.1619**
Temporal Understanding	EgoTempo	29.60	33.40	23.40	**34.40**
	Video-MME	50.70	61.30	59.96	**62.78**
	MVBench	47.79	58.58	**60.47**	60.08
	LongVideoBench	48.00	58.07	54.33	**58.65**
	QVHighlights-TimeLens	46.62	44.72	28.09	**51.72**
State Verification	StateBench-P	56.00	**65.40**	59.00	59.20
	StateBench-T	25.52	**33.46**	27.92	26.79
General	MMMU	56.86	**59.71**	44.29	53.05
	MMVet	62.02	62.48	46.51	**65.97**
	RealWorldQA	**76.08**	75.69	72.03	68.76
	MMBench-EN	88.45	88.20	86.90	**88.54**
	IFEval	81.52	**81.70**	63.59	70.61
	MMLU-Pro	**62.57**	61.30	37.57	54.20
	BFCL-v3	25.37	**25.75**	2.00	0.13
	LiveCodeBench v6	11.01	**26.65**	10.79	16.96

All results are reproduced under the DeepInsight evaluation protocol. Unless otherwise specified, all entries report the native benchmark metric (higher is better, in %). Exceptions include: † point-level F1 (%, higher is better); ‡ NaviTrace official semantic-aware normalized score (higher is better and may be negative); and § distance-based metrics, including VABench-trace trajectory RMSE (pixels) and ShareRobot-Trajectory discrete Fréchet distance (DFD), where lower is better. Bold and underlined values denote the best and second-best results. The blue column highlights Capek 0.5.

E Additonal Experimental Results

The compact 2B track shows a similar pattern under a different initialization and capacity regime. Capek 0.5-2B improves 30 of 34 matched rows, including all 9 Spatial Reasoning rows, all 10 Action Guidance rows, 4 of 5 Temporal Understanding rows, both State Verification rows, and 5 of 8 general-retention controls. The largest gains again appear on execution-facing outputs: VABench-point increases by 28.87 points, PIO by 24.59 points, NaviTrace by 54.67 points, VABench-trace RMSE decreases by 94.84 pixels, and ShareRobot-Trajectory DFD decreases by 0.3006. This capacity-separated result suggests that the proposed capability organization is not specific to the 35B-A3B backbone, but transfers to a smaller dense model as the same execution-oriented pattern of improvement.

F QUALITATIVE EMBODIEDBENCH ROLLOUT

Table 9: Selected EmbodiedBench replay sample.

Sample	`EB-ALF / long_horizon / episode-10`	**Family**	`long-horizon manipulation`
Instruction	Pick up knife, slice apple, put knife in bowl, heat slice of apple in microwave, put apple slice on table.	**Outcome**	20 steps / 0 invalid / success

Figure 7 shows a successful long-horizon EmbodiedBench episode. The model first resolves the required tool and object, slices the apple, stores the knife in a bowl, loads and heats the apple slice in the microwave, and finally places it on the dining table. The episode illustrates how the same execution-centric interface combines object grounding, state-changing manipulation, appliance interaction, and final task-state verification within one rollout.



Step 4 `action_id=160` `slice Apple` | **Step 6** `action_id=133` `put object in Bowl` | **Step 11** `action_id=133` `put object in Microwave` | **Step 14** `action_id=150` `turn off Microwave` | **Step 20** `action_id=133` `place Apple on Table`

Figure 7: **Task-critical EmbodiedBench keyframes.** Each selected frame is paired with the model action executed at that step. The full action trace contains 20 successful steps with no invalid action.

Table 10: Execution outcome summary for Figure 7.

Step(s)	Model action sequence	Outcome summary
1–4	Find knife, pick it up, find apple, slice apple.	All calls execute successfully; the apple is transformed into a sliced state while the knife remains in hand.
5–6	Find bowl, put down the held object.	The knife is placed in the bowl, clearing the hand for the apple slice.
7–14	Find and pick up apple, find microwave, open it, place apple inside, close it, turn it on, turn it off.	The apple slice is loaded into the microwave and the appliance interaction completes without invalid actions.
15–20	Open microwave, find and pick up apple, close microwave, find dining table, put down the held object.	The heated apple slice is retrieved and placed on the dining table; the final environment state satisfies the task.

Table 11: Complete model action trace for the EmbodiedBench episode. Every step returns `Last action executed successfully.`

Step	Action	Step	Action
1	`action_id=34`: find Knife	**11**	`action_id=133`: put apple in Microwave
2	`action_id=127`: pick up Knife	**12**	`action_id=144`: close Microwave
3	`action_id=22`: find Apple	**13**	`action_id=149`: turn on Microwave
4	`action_id=160`: slice Apple	**14**	`action_id=150`: turn off Microwave
5	`action_id=16`: find Bowl	**15**	`action_id=143`: open Microwave
6	`action_id=133`: put knife in Bowl	**16**	`action_id=22`: find Apple
7	`action_id=22`: find Apple	**17**	`action_id=107`: pick up Apple
8	`action_id=107`: pick up Apple	**18**	`action_id=144`: close Microwave
9	`action_id=38`: find Microwave	**19**	`action_id=18`: find DiningTable
10	`action_id=143`: open Microwave	**20**	`action_id=133`: place apple on DiningTable

G QUALITATIVE VIGIL ROLLOUTS

Table 12: Selected complex VIGIL replay sample.

Sample	`sample-627`	**Family**	`T2_search_then_interact`
Instruction	Find the toaster, turn it on, then report success.	**Score**	0.9543 / success

Figure 8 shows a representative VIGIL episode in which the model first approaches the table, identifies the toaster among nearby objects, and repeatedly attempts to activate it. Several early activation calls do not change the task state; in the rollout, the model attributes these failures to being slightly out of interaction range or to imprecise targeting. This distance diagnosis is inferred by the model from the unchanged visual and task-state evidence, rather than returned as an explicit environment message. The model then moves closer, successfully toggles the toaster at Step 8, and reports success after the activated state becomes visible. This trajectory highlights a multi-step recovery behavior: the model does not stop after the first failed interaction, but continues to refine its position and action until the task state is achieved.



Figure 8: **Step-by-step VIGIL rollout.** Each frame is paired with the exact model function call for that observation. The successful activation occurs at Step 8; Step 10 closes the episode with a matching success report.

Table 13: Execution outcome summary for the function calls in Figure 8.

Step(s)	Outcome summary
1–2	The agent moves toward the table; no target interaction has been credited yet.
3–5	The calls are aimed at the toaster, but no activation is credited. The model infers from the unchanged state that it may still be too far away or imprecisely targeted; the environment does not expose this distance explanation as feedback.
6	The call appears to land away from the toaster, so no activation is credited.
7	The agent moves closer after the failed activation attempts.
8	The activation call hits the Toaster and the verifier marks the required object state as reached.
9–10	The state remains reached; the final `report(success)` matches the world state.

H QUALITATIVE STATIC QA ROLLOUTS

Each static QA example is shown as a paired visualization and model rollout. The selected samples cover spatial reasoning, point grounding, manipulation trajectory prediction, trace navigation, temporal video understanding, and general chart reading while keeping the evidence close to the corresponding image.

Physical-state verification: `StateBench-P / PSVMIX_21d9464a`



Across the two chronological observations, the model tracks the lighter's final state and returns the correct state label.

Question. Based on observations, is the lighter on or off?
Ground truth. `off`.

```
The user wants to know the state of the lighter.

1.  **Analyze the first image:** I see a first-person perspective of a character holding
      two items. The left hand is holding a green lighter. The top of the lighter is
      emitting a flame. The right hand is holding a gun. There is also a fire in the
      fireplace.
2.  **Analyze the second image:** The character is still holding the green lighter in
      the left hand. The top of the lighter is now glowing red/orange, indicating it's
      hot or just used, but crucially, there is no flame coming out of it. The hand
      holding the lighter is moving away/down. The right hand is holding the gun,
      pointed downwards.
3.  **Synthesize observations:** In the first image, the lighter is clearly on (flame
      visible). In the second image, the flame is gone. The question asks "Based on
      observations, is the lighter on or off?". Looking at the progression, the lighter
       was lit and is now extinguished. The final state shown in the second image is "
      off". Wait, let me re-evaluate. The question is singular. "is the lighter on or
      off?".
    *   Image 1: Lighter is ON (flame visible).
    *   Image 2: Lighter is OFF (no flame, tip is red/hot).
    *   Usually, these questions refer to the final state or the general action. Let's
      look at the second image more closely. The lighter is held in the left hand. The
      tip is red. There is no flame. Therefore, the lighter is off.

Let's double check.
Image 1: Green lighter, flame is visible at the top.
Image 2: Green lighter, no flame, tip is red.
Conclusion: The lighter is off.
</think>

off
```

Task-state verification: `StateBench-T / B1K_CPV_8d5e714d`



The model correctly tracks the multi-object packing state and identifies opening the refrigerator door as the next action, but underestimates completion (75% vs. 83.33%) because of an inconsistent goal-condition count.

Question. Task: Put both apple halves, the club sandwich, and the chocolate chip cookie from the chopping board on the kitchen countertop into the packing box on the countertop. Then take the bottle of tea out of the refrigerator, put it into the same box, and close the refrigerator when you're done. Estimate completed task percentage and predict the immediate next action.
Ground truth. `{"progress": 83.33, "next_action": "open the fridge door"}`.

```
The task is to complete all of the required packing conditions.

1. **Identify the goal conditions:** There are six conditions that must be satisfied:
   * The apples must be inside the packing box.
   * The club sandwich must be inside the packing box.
   * The chocolate chip cookie must be inside the packing box.
   * The bottle of tea must be inside the packing box.

2. **Analyze the current state:** Both apple halves, the club sandwich, and the
      chocolate chip cookie are already inside the packing box. However, the bottle of
      tea has not yet been placed inside the packing box.

3. **Evaluate the progress:** This represents a progress of 3 out of 4, corresponding to
       83.33% completion. The only remaining unsatisfied condition is that the bottle
      of tea must be placed inside the packing box.

4. **Determine the next action:** The bottle of tea is inside the refrigerator, which is
       currently closed. Therefore, the immediate next step is to open the refrigerator
       door so the bottle can be retrieved. After placing the bottle of tea inside the
      packing box, the refrigerator will need to be closed again to ensure that all six
       goal conditions are satisfied.

Let's double-check:
* First apples inside the packing box: satisfied.
* Club sandwich inside the packing box: satisfied.
* Chocolate chip cookie inside the packing box: satisfied.
* Bottle of tea inside the packing box: not yet satisfied.

Conclusion: three of the four conditions are satisfied, so the task is 75% complete. The
       next action is to open the refrigerator door.
</think>

{"progress": 75, "next_action": "open the refrigerator"}
```

Spatial reasoning: `OpenEQA / 1122`



The model searches across bathroom views and identifies the shower shelf as a feasible place for shampoo.

Question. Where can I place a bottle of shampoo while I shower?
Ground truth. On the tray in the shower area or on the bathtub ledge.

```
The user wants to know where to place a bottle of shampoo while showering. I need to
      look for a shower area and identify any shelves or holders.

1.  **Scan the images for a shower area:**
    *   Images 1-5 show the toilet and part of the bathtub.
    *   Images 6-9 show a shower curtain and a shower head.
    *   Image 8 specifically shows a small glass shelf in the shower area.
    *   Image 9 shows the shower head and the shelf again.
    *   Image 10 shows the shower area with a glass shelf holding some items (possibly
      shampoo bottles).
    *   Image 14 shows the shower curtain and a grab bar.
    *   Image 19 shows the bathtub and shower curtain.

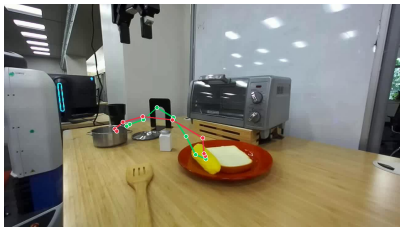
2.  **Identify the specific location:**
    *   Looking closely at Image 8, 9, and 10, there is a small, clear glass shelf
      attached to the wall inside the shower/tub area.
    *   In Image 10, there are actually small bottles sitting on this shelf, confirming
      it's used for toiletries like shampoo.

3.  **Formulate the answer:**
    *   State that there is a shelf.
    *   Describe its location (in the shower/tub area, on the wall).
    *   Mention that it can hold the bottle.

4.  **Refine the answer:** "You can place a bottle of shampoo on the small glass shelf
      located inside the shower area, attached to the wall." This is accurate based on
      the visual evidence.
</think>

You can place a bottle of shampoo on the small glass shelf located on the wall inside
      the shower area.
```

Grounding and trajectory: `VABench_trace / fsd_pointdroid_4060`



The model identifies the yellow object and silver pot, then emits an eight-point object-centric trajectory.

Question. Put the yellow object into the silver pot; output eight ordered object-centric waypoints.
Ground truth. `[(652,500), (619,486), (588,427), (547,366), (494,334), (450,373), (406,388), (396,394)]`

```
Full Rollout:
The user wants me to identify 8 waypoints to guide a robot to perform the task "Put the
      yellow object into the silver pot".

1.  **Identify the objects:**
    *   **Yellow object:** This is the corn on the cob on the red plate.
    *   **Silver pot:** This is the metal bowl/pot to the left of the plate.

2.  **Determine the trajectory:**
    *   The robot needs to approach the yellow object (corn).
    *   It needs to grasp the yellow object.
    *   It needs to lift the yellow object.
    *   It needs to move the yellow object over the silver pot.
    *   It needs to place the yellow object into the silver pot.

3.  **Select waypoints:**
    *   **Point 1:** Initial approach to the yellow object. Let's say, near the corn.
      [508, 682]
    *   **Point 2:** Grasping the yellow object. [500, 690]
    *   **Point 3:** Lifting the object slightly. [510, 600]
    *   **Point 4:** Moving towards the pot. [400, 550]
    *   **Point 5:** Positioning over the pot. [280, 550]
    *   **Point 6:** Lowering into the pot. [280, 580]
    *   **Point 7:** Placing the object inside. [280, 600]
    *   **Point 8:** Final adjustment or release. [290, 590]

    Wait, let me refine the points to be more precise and follow a logical path.
    *   Start near the corn: [508, 682]
    *   Grasp the corn: [505, 685]
    *   Lift up: [515, 550]
    *   Move left towards the pot: [400, 500]
    *   Move further left and slightly down towards the pot opening: [300, 530]
    *   Position directly over the pot: [270, 540]
    *   Lower into the pot: [270, 580]
    *   Release/adjust: [280, 590]

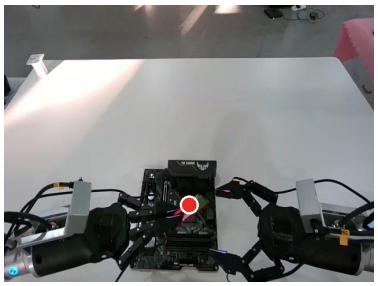
    Let's re-evaluate the coordinates.
    *   Corn is roughly at x=500, y=680.
    *   Pot is roughly at x=270, y=580.

    Let's try:
    1.  Approach corn: [508, 682]
    2.  Grasp corn: [500, 690]
    3.  Lift corn: [505, 620]
    4.  Move towards pot (mid-air): [380, 500]
    5.  Position over pot: [300, 520]
    6.  Lower into pot: [285, 560]
    7.  Place inside: [285, 580]
    8.  Final placement: [290, 575]

    This looks like a reasonable 8-point trajectory.
</think>

'''json
[
  {"point_2d": [508, 682], "label": "waypoint 1"},
  {"point_2d": [505, 672], "label": "waypoint 2"},
  {"point_2d": [502, 600], "label": "waypoint 3"},
  {"point_2d": [425, 520], "label": "waypoint 4"},
  {"point_2d": [350, 505], "label": "waypoint 5"},
  {"point_2d": [300, 530], "label": "waypoint 6"},
  {"point_2d": [280, 560], "label": "waypoint 7"},
  {"point_2d": [285, 570], "label": "waypoint 8"}
]
'''
```

Pointing: `PIO / 47`



The model grounds an execution handoff by selecting the grasp point where the receiving gripper should act.

Question. The left gripper is passing the device to the right gripper. Where should the right gripper act to grasp the device?
Ground truth. Valid grasp region shown in green; model point shown in red; Capek score: `acc=1.0`.

```
The user wants to identify a point where the right gripper should grasp the device.
1.  **Identify the device:** The device is the red object being held by the left gripper
      . It looks like a small, rectangular component, possibly a stick of RAM or a
      similar computer part.
2.  **Identify the left gripper:** This is the robotic arm on the left side of the image
      , holding the red device.
3.  **Identify the right gripper:** This is the robotic arm on the right side of the
      image.
4.  **Determine the grasp point:** The right gripper needs to grasp the red device to
      take it from the left gripper. The most logical point for the right gripper to
      grasp is the center or a stable part of the red device itself.
5.  **Locate the red device:** It's positioned roughly in the center, slightly below the
       horizontal midline, being held by the left gripper.
6.  **Select a point:** A point on the red device, specifically near where the right
      gripper would approach, is appropriate. The red object is centrally located
      between the two grippers. A point on the red object itself, perhaps slightly
      towards the right side of it to align with the right gripper, would be good. Let'
      s aim for the center of the red object.

Coordinates estimation:
- The red object is roughly at x=490, y=715.
- The right gripper is approaching from the right.
- So, a point like [495, 720] seems reasonable for where the right gripper should grasp.
</think>

'''json
[
  {"point_2d": [501, 722], "label": "point_1"}
]
'''
```

Trace navigation: `NaviTrace / 9d8a...`



The model chooses the right tunnel and produces a curved path that stays close to the official reference trajectory.

Question. Take the right tunnel.
Ground truth. Official best-matching trajectory shown in green; model trajectory shown in red; `dfd=0.045`.

```
The user wants me to plan a trajectory for a Human to "Take the right tunnel".
1.  **Identify the current position:** The image shows a gravel path leading towards a
      rock face with two tunnels. The starting point should be near the bottom center.
2.  **Identify the goal:** The task is to "Take the right tunnel". Looking at the rock
      face, there are two openings. The one on the right is the target.
3.  **Determine the path:**
    *   The path starts on the gravel road.
    *   The road curves slightly or leads straight towards the tunnels.
    *   To enter the *right* tunnel, the person needs to aim for the opening on the
      right side of the rock face.
    *   The path seems to split or widen slightly as it approaches the tunnels.
    *   I need to generate points moving from the bottom center towards the right-hand
      tunnel entrance.

Let's pick points:
*   Start: [500, 900] (Bottom center)
*   Mid 1: [520, 750] (Moving up the path)
*   Mid 2: [550, 600] (Approaching the junction/tunnel area)
*   Mid 3: [600, 550] (Aiming for the right tunnel entrance)
*   Goal: [650, 520] (Entering the right tunnel)

Wait, looking closely at the image, the path leads towards two tunnels. The left tunnel
      is smaller/further back. The right tunnel is larger and closer to the foreground
      on the right. The path seems to go towards the space between them but leans
      slightly right towards the bigger opening. Let's refine the points to aim
      specifically for the right opening.

Refined points:
*   Start: [480, 920]
*   Point 2: [500, 780]
*   Point 3: [530, 650]
*   Point 4: [580, 580] (Approaching the entrance)
*   Point 5: [630, 550] (Entering the right tunnel)

Let's look at the coordinates again.
The bottom center is roughly x=500, y=950.
The right tunnel entrance is roughly x=650, y=520.
The path is gravel.

Let's trace a line.
[486, 946] -> [512, 800] -> [538, 680] -> [582, 600] -> [625, 550] -> [650, 530]

This looks reasonable. The trajectory should end inside or at the entrance of the right
      tunnel.
</think>

'''json
[
  {"point_2d": [508, 913], "label": "waypoint 1"},
  {"point_2d": [516, 805], "label": "waypoint 2"},
  {"point_2d": [535, 690], "label": "waypoint 3"},
  {"point_2d": [563, 615], "label": "waypoint 4"},
  {"point_2d": [608, 570], "label": "waypoint 5"},
  {"point_2d": [640, 555], "label": "waypoint 6"}
]
'''
```

Video and temporal: `EgoTempo / grapes`



The model identifies what happens immediately after the grapes are weighed by following the frame sequence.

Question. What happens after the person weighs the grapes?
Ground truth. The person puts the grapes into a plastic bag and then places it in the shopping basket.

```
The user wants to know what happens after the person weighs the grapes.

1.  **Identify the grape weighing scene:**
    *   At 00:05, the person puts green grapes on the scale.
    *   At 00:06, the scale shows the weight.
    *   At 00:09, the person takes the grapes off the scale.

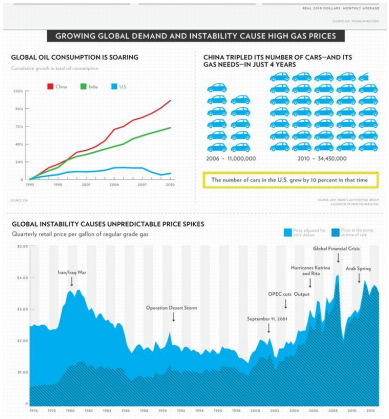
2.  **Observe the immediate next action:**
    *   Right after taking the grapes off the scale (00:09), the person is seen holding
      a clear plastic bag.
    *   At 00:10, they put the grapes into the plastic bag.
    *   At 00:13, the person is holding the bag with the grapes inside.
    *   At 00:14, the person puts the bag with the grapes into a red shopping basket.

3.  **Synthesize the answer:**
    *   The person takes the grapes off the scale.
    *   They put them into a plastic bag.
    *   They place the bag into their shopping basket.

4.  **Formulate the final response:** The person puts the grapes into a plastic bag and
      then places the bag into a shopping basket.
</think>

The person puts the grapes into a plastic bag and places it in the shopping basket.
```

General: `InfoVQA_VAL / 95160`



The model locates the 2010–2012 price-spike region and answers that the relevant cause is the Arab Spring.

Question. What caused unpredictable price spikes between 2010 and 2012?
Ground truth. Arab Spring.

```
The user wants to identify the cause of unpredictable price spikes between 2010 and 2012
       based on the infographic.

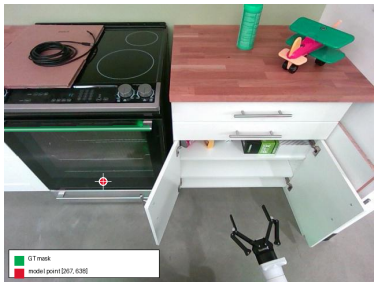
1.  **Scan the infographic for sections related to price spikes.** I see a section
      titled "GLOBAL INSTABILITY CAUSES UNPREDICTABLE PRICE SPIKES". This looks very
      promising.
2.  **Look at the chart in that section.** It's a bar/area chart showing "Quarterly
      retail price per gallon of regular grade gas" from 1976 to 2012.
3.  **Identify the specific time period.** The user is asking about 2010-2012.
4.  **Locate the year 2010-2012 on the x-axis.** I see labels for 2010, 2011, 2012.
5.  **Look for labels pointing to spikes in that area.** There is an arrow pointing to a
       spike around 2011/2012 labeled "Arab Spring".
6.  **Verify the context.** The question asks for the "reason". The section title is "
      GLOBAL INSTABILITY CAUSES UNPREDICTABLE PRICE SPIKES". The specific label
      pointing to the 2010-2012 spike is "Arab Spring".
7.  **Formulate the answer.** The user requested a single word or phrase. "Arab Spring"
      fits perfectly.
</think>

Arab Spring
```

H.1 STATIC QA FAILURE CASES

The examples below show representative failures under the same static-QA evaluation interface. In each case, the rollout remains interpretable: the model identifies plausible visual evidence, but commits to an incorrect execution target or spatial relation. These cases complement the successful samples above by showing where execution-facing supervision still exposes brittle decisions.

Failure: `PIO / 432`



The model correctly identifies the oven but places the action point on the lower door surface rather than the annotated handle.

Question. Where should I apply force to open the oven door?
Ground truth. GT mask in green; parsed model point in red, `[267, 638]`; `acc=0.0000`.

```
The user wants to find the location to apply force to open the oven door.
1.  **Identify the oven door:** It's the large black rectangular area on the left side,
      below the stovetop.
2.  **Identify the handle:** The oven door typically has a handle used for opening.
      Looking at the oven, there is a horizontal silver bar near the bottom edge of the
       door.
3.  **Determine the force application point:** To open a standard oven door like this,
      one would pull the handle. The handle is located roughly in the middle of the
      lower part of the door.
4.  **Select a point:** A point on the handle or just below it on the door panel would
      be appropriate. The handle is clearly visible as a horizontal bar. Let's pick a
      point on the handle itself or slightly below it on the door frame where the
      handle is attached. The handle is roughly at x=250, y=640. Wait, looking closer,
      the handle is the silver bar near the bottom of the black door. Let's aim for the
       center of that handle.
    *   The oven door is the black part.
    *   The handle is the silver bar near the bottom of the black door.
    *   Coordinates for the handle area: x is around 250-300, y is around 650.
    *   Let's refine. The handle is the silver bar. A good point to pull would be the
      center of that bar.
    *   Looking at the image, the handle is at the bottom of the oven door.
    *   Point: [260, 655] seems like a good spot on the handle.

Let's re-evaluate. The question asks "Where should I apply force to open the oven door
      ?".
The oven door is the large black panel.
The handle is the silver horizontal bar at the bottom of the oven door.
So, applying force to the handle is the correct action.
I will place a point on the handle.
Coordinates: x=264, y=652.

Wait, let's look at the handle again. It's the silver bar running horizontally near the
      bottom edge of the black glass.
Yes, [264, 652] is on that handle.

Let's check the cabinet door. The cabinet door is open on the right. The question is
      specifically about the *oven* door.
The oven is the appliance on the left.
So the point must be on the oven door handle.

Final check of coordinates:
x=264 is roughly the middle horizontally for the oven.
y=652 is near the bottom of the oven door where the handle is.
This looks correct.
</think>

'''json
[
  {"point_2d": [267, 638], "label": "point_1"}
]
'''
```

Failure: `VABench-point / fsd_pointbridge_94165`



The model finds the toy rabbit but grounds the destination near the drawer front instead of the top surface of the drawer.

Question. Move the toy rabbit on top of the drawer.
Ground truth. GT mask in green; parsed model point in red, `[237, 868]`; `acc=0.0000`.

```
The user wants me to move the toy rabbit to the top of the drawer.
1.  **Identify the object:** The object is the grey toy rabbit with white ears and a
      pink nose located near the bottom center of the image.
2.  **Identify the target location:** The instruction says "on top of the drawer".
      Looking at the image, there is a transparent plastic surface in the bottom left
      corner which looks like the top of a drawer or a shelf. The rabbit is currently
      on the wooden table. I need to move it to that transparent surface.
3.  **Determine the point:** I need to pick a point on the transparent surface where the
       rabbit should be placed. A good spot would be roughly in the center of that
      visible transparent area. Looking at the bottom left, the transparent surface is
      visible. Let's aim for a point on that surface. Wait, looking closer, the "drawer
      " might refer to the white box-like structure on the left, but usually "drawer"
      implies a sliding compartment. The transparent plastic sheet at the bottom looks
      like a drawer front or a shelf. Let's assume the transparent area is the "drawer"
       top. Or perhaps the white box is a drawer? No, the prompt says "on top of the
      drawer". The transparent part at the bottom left looks like the top of a drawer
      unit. Let's aim for a point on that transparent surface.
4.  **Refine the point:** The rabbit is at [570, 680]. The transparent surface is below
      and to the left. A point like [200, 850] seems appropriate for "on top of the
      drawer". Let's look again. Is there another drawer? The white box on the left has
       a red stripe. It looks like a container. The transparent part is definitely a
      surface. Let's assume the transparent surface is the target.
5.  **Final Point Selection:** I will select a point on the transparent surface visible
      in the bottom left corner. Let's say around x=200, y=850. Wait, looking at the
      coordinates, y increases downwards. So y=850 is near the bottom edge. x=200 is
      towards the left. This seems like a valid spot on the transparent surface.

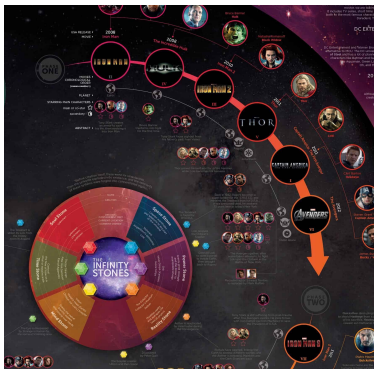
Let's re-evaluate "on top of the drawer". The white box on the left looks like a drawer
      unit or a container. The transparent part is likely the front of a drawer or a
      shelf. Let's assume the transparent surface is the intended target.
Point: [220, 860] seems reasonable. It's on the clear plastic surface.

Let's look at the object again. It's a grey plush toy, looks like a mouse or rabbit.
Target: "on top of the drawer".
The transparent surface at the bottom left is the most likely candidate for "drawer top"
       or a shelf above a drawer.
Let's pick a point there. [250, 880].

Actually, looking at the layout, the transparent part is likely a shelf or the top of a
      drawer unit below the main table.
Let's place the point there.
</think>

```json
[
  {"point_2d": [237, 868], "label": "point"}
]
```
```

Failure: `InfoVQA_VAL / 93846`



The model reads the Phase One release label and ignores the infographic's Roman-numeral chronological ordering.

Question. In phase 1, which movie comes first in the chronological order?
Ground truth. GT: `Captain America: The First Avenger`; parsed answer: `Iron Man`.

```
The user wants to identify the first movie in Phase 1 based on the provided infographic.

1.  **Locate "Phase One":** I need to scan the image for the text "PHASE ONE". I see it
      in the upper-left quadrant of the circular timeline.
2.  **Understand the layout:** The timeline is circular, but there's a specific section
      labeled "PHASE ONE". It seems to flow clockwise starting from the top.
3.  **Identify the first movie:** Looking at the start of the Phase One section, there's
       a circle labeled "IRON MAN" with the Roman numeral "I" below it. Next to it,
      there's text saying "2008 Iron Man".
4.  **Verify chronological order:** The text explicitly mentions "MOVIES CHRONOLOGICAL
      ORDER (roman numbers)". The first movie has the Roman numeral "I". This confirms
      "Iron Man" is the first one.
5.  **Formulate the answer:** The question asks for a single word or phrase. "Iron Man"
      is the correct answer.
</think>

Iron Man
```