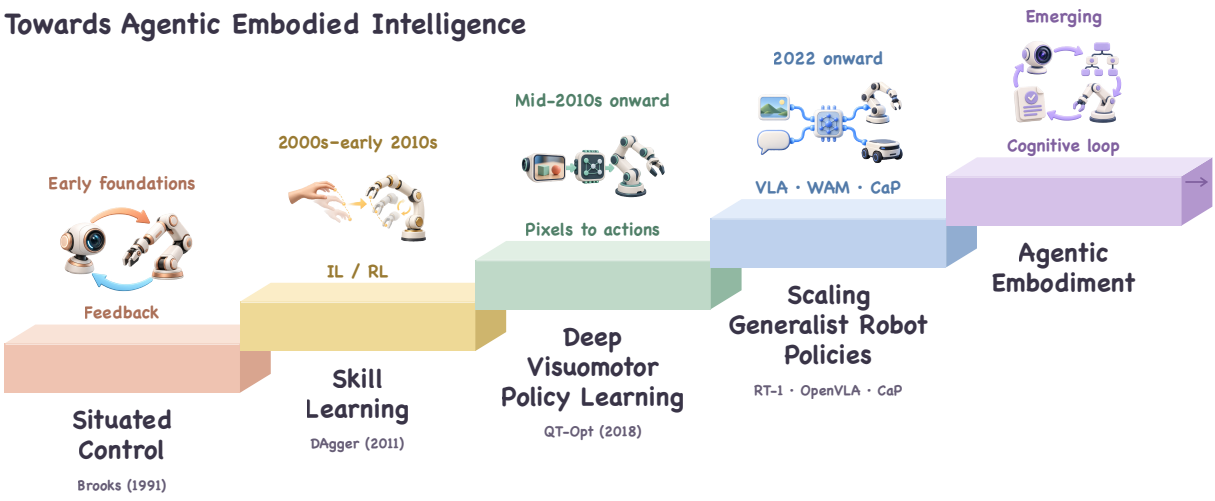
Towards Agentic Embodied Intelligence

AEI Team

Abstract

Embodied Intelligence represents the ultimate frontier of bridging artificial intelligence with physical reality to achieve general-purpose physical agency. Recently, the infusion of multimodal foundation models has catalyzed an end-to-end sensorimotor paradigm, translating internet-scale semantic representations and human intents into motor actions. However, the prevailing monolithic paradigm treats continuous physical interaction as an unconstrained, open-loop sequence generation task. Consequently, it suffers from catastrophic compounding covariate drift, temporal attention dispersion over long horizons, and a complete absence of in-situ error recovery, fundamentally confining current systems to fragile, human-dependent conditional autonomy. We argue that transcending this plateau cannot be achieved by merely scaling feedforward policy weights; rather, the decisive breakthrough demands a fundamental paradigm shift toward **Agentic Embodied Intelligence** (AEI). In this survey, we examine embodied systems through an agentic lens, systematically characterizing the paradigm shift toward AEI across models, infrastructure, safety, data, environments, and evaluation. We formalize a cognitive closed loop: elucidating how physical grounding, deliberative planning, motor execution, and in-situ reflection dynamically interlock to bridge the chasm between abstract reasoning and continuous physical dynamics. Recognizing that the community remains in the nascent formative stage of building AEI, we critically dissect the prevailing open challenges across physical fidelity, real-time latency, and trustworthy deployment. We envision this work serving as a roadmap to guide future exploration toward resilient, general-purpose Physical AI in the open world.

Project Page: https://agenticei.github.io/AEI-Survey-page/



Contents

1 Introduction **4**
- 1.1 Development of Embodied Systems 4
- 1.2 Ladder of Embodied Autonomy 6
- 1.3 Scope and Contribution 8

2 Conceptual Framework **9**
- 2.1 System Organization 9
- 2.2 Model Cognitive Loop 9

3 Perception and Grounding **11**
- 3.1 Semantic Grounding 11
- 3.2 Spatial Grounding 13
- 3.3 Physical Grounding 15
- 3.4 From Passive Perception to Active Perception 16

4 Reasoning and Planning **19**
- 4.1 Problem Formulation 19
- 4.2 Hierarchical Task Decomposition 20
- 4.3 Plan Search and Candidate Selection 21
- 4.4 Plan Representation 22

5 Action Execution **23**
- 5.1 Observation and Task Conditioning 24
- 5.2 Action Representation 25
- 5.3 Action Generation 26
- 5.4 Control Execution 27

6 Closed-Loop Evolution **28**
- 6.1 Evolution Targets 29
- 6.2 How Coding Agents Implement Closed-Loop Improvement 30
- 6.3 Embodied Closed-Loop Evolution: Current Approaches 30
- 6.4 From Component Loops to a Complete Embodied Evolution Pipeline 33

7 Computing Infrastructure **34**
- 7.1 From End-to-End to Agentic: The Infrastructure Requirement 34
- 7.2 Deployment across Device, Edge, and Cloud 35
- 7.3 Agent Runtimes and Interfaces 36
- 7.4 Hardware and Embodiments 37

8 Safety and Trustworthiness **37**
- 8.1 Scope and Safety Requirements 38
- 8.2 Safety-Critical Perception and Risk Awareness 39
- 8.3 Safe Decomposition and Execution of Long-Horizon Tasks 39
- 8.4 Safe Policy and Runtime Enforcement 41
- 8.5 Safe Reflection, Memory, and Evolution 41
- 8.6 Evaluation and Assurance Gaps 42

9 Data Ecosystem **44**
- 9.1 From the Data Pyramid to Agentic Embodied Experience 44
- 9.2 Generating Agentic Embodied Experience 47
- 9.3 Closing the Loop: The Agentic Data Flywheel 48

10 Simulators, Benchmarks and Evaluation Protocols **50**

10.1 Physics-Semantic Hybrid Simulation Foundations 50
10.2 AEI Benchmark Spectrum: The Cognitive Capability Ladder 52
10.3 Decoupled Evaluation Metrology & Verifiable Trustworthiness 55

11 Open Challenges and Future Horizons 57
11.1 Perception and Grounding 57
11.2 Reasoning and Planning 58
11.3 Action Execution 60
11.4 Closed-loop Evolution 60
11.5 Computing Infrastructure 61
11.6 Safety and Trustworthiness 61
11.7 Agentic Data Ecosystem 62
11.8 Simulators, Benchmarks and Evaluation Protocols 62

12 Conclusion 64

Authors 64

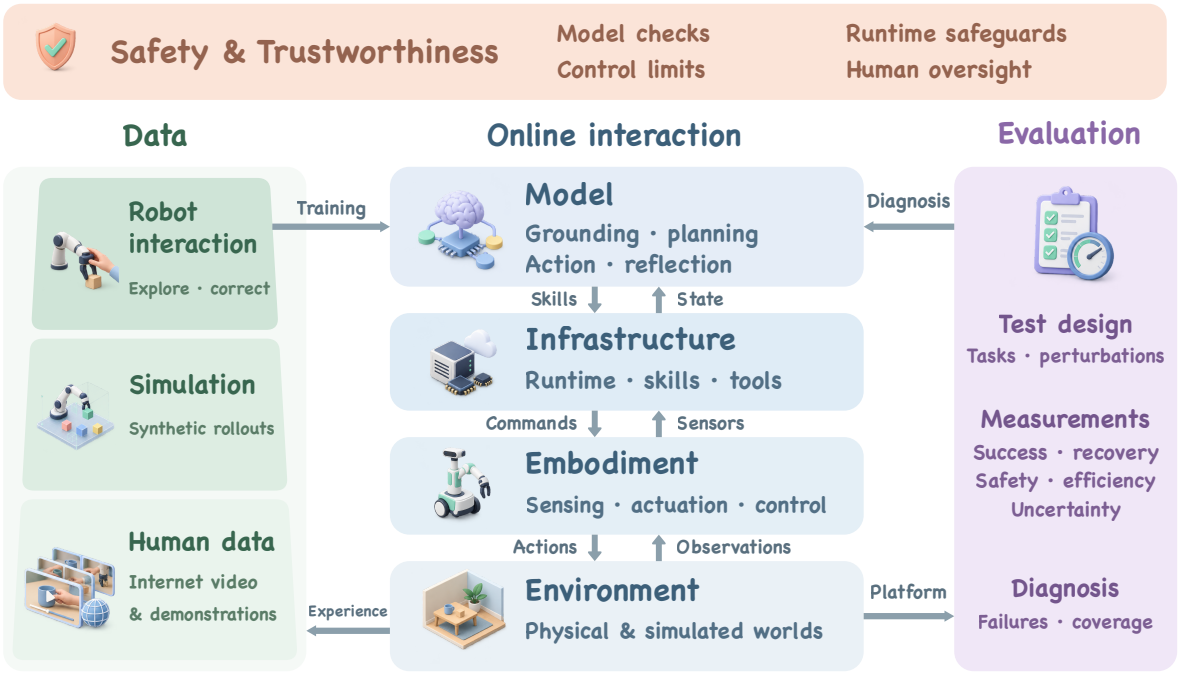


Figure 1 System-level relationships in agentic embodied intelligence. The central embodied core is the online interaction loop: the model grounds observations, plans, acts, and reflects; infrastructure translates its skills into commands for the robot; and the robot interacts with the physical or simulated environment, returning sensor data and state through the infrastructure to the model. The environment serves two connected roles: it generates interaction data and simulated rollouts for training, and it provides the platform for evaluating models through tasks and perturbations. Evaluation results and failure diagnoses then inform model development. Safety and trustworthiness cut across every part of this system, from model checks and control limits to runtime safeguards and human oversight, and form a final, essential requirement for practical deployment.

1 Introduction

The rapid advancement of large language models (LLM) and multimodal foundation models has catalyzed a paradigm shift in artificial intelligence, moving beyond purely digital information processing toward agents that can perceive, reason, and act in the physical world, motivating the development of embodied intelligence. A prevalent but reductionist view frames this emerging field as simply *installing large models onto robots*, yet the core principle of embodied intelligence is far more fundamental: it centers on the formation of a closed perception, cognition, and action loop embedded in the physical world, where intelligent behavior emerges from dynamic, reciprocal interactions between an agent, its body, and the environment, rather than from disembodied symbolic computation alone [54, 68, 91, 350]. The notion of "embodiment" emphasizes that intelligence is inherently constrained by the agent's physical morphology, sensorimotor capabilities, actuator limits, and the laws of physics, and that knowledge and skills are acquired through active exploration, physical manipulation, and environmental feedback rather than passive data ingestion [43, 123, 209, 272].

1.1 Development of Embodied Systems

The rise of embodied intelligence is often misframed as a sudden breakthrough spurred solely by recent large foundation models. In reality, its intellectual lineage spans several decades, evolving through successive paradigm shifts driven by the cross-pollination of cybernetics, cognitive science, robotics, and artificial intelligence. This long developmental trajectory traces a consistent intellectual arc: a steady departure from the classical view of intelligence as disembodied symbolic computation, toward a foundational consensus that intelligent behavior emerges from the closed-loop, reciprocal interactions among an agent's physical body, its

sensorimotor systems, and the constraints of the physical environment. Retracing this history is essential for contextualizing today's rapid advances: it illuminates how long-standing theoretical insights have been gradually operationalized into engineering systems, how each technological wave has redefined the field's core questions, and why the ongoing convergence of foundation models and robotics represents both a natural culmination of prior work and a qualitative leap toward general-purpose physical intelligence.

Situated-based Control and Sensorimotor Development. Early embodied robotics challenged the idea that intelligent action must be produced by a single central model that first constructs a complete symbolic description of the world. Brooks's layered control system organized behavior around ongoing interaction with the environment, and his later "Intelligence without Representation" argued that robust competence could arise from situated perception–action coupling rather than elaborate internal world models [68]. Developmental robotics made the sensorimotor loop itself an object of study: an agent's perceptual and action capabilities were treated as developing through interaction, rather than being supplied only as fixed modules [250]. This early period is a conceptual and control-architecture foundation, not an embryonic version of today's pretrained embodied models. The systems were typically designed around particular robots, sensors, and behavioral requirements. Its lasting contribution is a framing principle: embodiment, action, and environmental feedback help determine what an agent can perceive and learn. Later data-driven methods changed how behaviors were acquired, but did not make physical interaction or feedback a new invention.

From Programmed Behaviors to Robot Skill Learning. During the 2000s and early 2010s, robot learning consolidated demonstration-based learning [42] and reinforcement learning [198] as systematic alternatives to specifying every behavior by hand. In physical robotics, apprenticeship learning enabled autonomous helicopter aerobatics from expert demonstrations, while policy search optimized robot motor primitives through interaction [22, 200]. DAgger addressed the compounding errors that arise when an imitation policy visits states absent from its demonstrations by iteratively collecting expert labels on states induced by the learner [308]. The learned unit was usually a task- or embodiment-specific skill: a policy for a defined manipulation, locomotion, or control problem, rather than a general-purpose model conditioned on open-ended instructions.

Deep Visuomotor Policies Learning. From roughly 2015, deep learning made it practical to learn visuomotor policies from pixels and robot experience. Previous works [214] demonstrated end-to-end training of deep visuomotor policies, and subsequent grasping work emphasized large-scale data collection and closed-loop, off-policy learning on physical robots [182, 215, 289]. In parallel, sim-to-real transfer became an explicit research problem; domain randomization, for example, varied simulated appearance and dynamics to improve transfer of visual policies to real robots [348]. These results expanded the scale and learning machinery for robot skills, but success remained tied to particular action spaces, tasks, data regimes, and robot platforms. At the same time, a distinct embodied infrastructure emerged around interactive simulators and standardized tasks. AI2-THOR [201] provided household scenes in which agents could navigate and interact with objects; Habitat [309] emphasized fast, configurable embodied-agent simulation; and ALFRED [324] joined egocentric perception with compositional natural-language instructions for long-horizon household tasks. Works [37, 97] on embodied question answering and vision-and-language navigation similarly made an agent's need to move, observe, and act part of the task definition. Later platforms [100, 163, 339]. The physical-policy and simulator-benchmark lines advanced concurrently; neither is a simple successor to the other. Together they made embodied tasks more measurable and accelerated iteration, but benchmark performance in a simulator did not establish robust deployment on physical robots. The main change was that embodied intelligence increasingly became a reproducible learning-and-evaluation problem, not only a collection of robot-specific control demonstrations.

Scaling Generalist Robot Policies. While large scale pretraining demonstrates impressive generalizable capability in natural language processing, researchers began to study whether robot skills could be learned as a shared policy from heterogeneous tasks, demonstrations, and embodiments, rather than as isolated controllers. Early instantiations [66, 103] established the feasibility of transferring web-scale visual-linguistic priors into physical control. This trajectory crystallized into the Vision-Language-Action (VLA) paradigm [60, 64, 196] by transferring web-scale visual-semantic knowledge into robot action prediction. In parallel, world-action models (WAMs) [147, 195, 283] add predictive structure to action learning. The motivation is to make temporal

Table 1 Taxonomy of Embodied Autonomy Levels.

Level	Level Name	Capability Definition	Human Role
L0	**Direct Teleoperation**	Direct mapping of continuous human motions to joint commands.	Physically controls every action frame in real-time.
L1	**Assisted Primitives**	Executes single closed-loop atomic skills, while task decomposition and sequencing remain manual.	Decomposes tasks and sequentially issues atomic commands.
L2	**Closed-set Autonomy**	Autonomously executes a complete, predefined single task in controlled settings.	Maintains constant visual line-of-sight; ready for immediate physical takeover.
L3	**Conditional Autonomy**	Comprehends natural language; accomplishes multi-step composite tasks under moderate variations.	Not required for continuous monitoring, but must intervene upon request within a time budget.
L4	**High Domain Autonomy**	Fully autonomous closed-loop execution; features long-horizon multi-step workflows.	Zero on-site physical intervention; manages macro missions and handles asynchronous exception tickets.
L5	**Full General Autonomy**	Operates unbounded open worlds with human-level commonsense reasoning, tool-use transfer, active perception, and lifelong self-evolution.	Requires zero technical expertise; interacts naturally via high-level goals and preferences.

consequences available to action selection: instead of only asking which action matches the current input, the model can learn which action is expected to produce a useful next state. Recently, with frontier models exemplified by next-generation agents [38, 281] demonstrating unprecedented software engineering proficiency, test-driven synthesis, and iterative traceback debugging, the Code-as-Policies (CaP) [234] paradigm has commanded widespread renewed attention as a scalable bridge between abstract cognition and physical execution. Fundamentally, Cap reconceptualizes robot policy generation not as continuous motor torque regression or unstructured natural language dialogue, but as programmatic code synthesis. Under this formulation, a foundation model functions as a real-time software engineer: given a complex, long-horizon user instruction and real-time environmental context, it autonomously generates executable scripts that orchestrate a suite of underlying perception APIs and motor control primitives.

Towards Open-World Agentic Embodiment. A distinct, still-emerging direction treats the robot not merely as the endpoint of a policy call, but as an agent that repeatedly interprets a goal, gathers task-relevant evidence, selects among skills and tools, observes physical outcomes, and changes its plan when reality diverges from expectation. Current demonstrations generally operate over bounded task sets, finite skill libraries, short deployments, or controlled environments. They do not yet provide a general guarantee of calibrated perception, safe long-horizon execution, recovery from unanticipated failures, and continual improvement across arbitrary homes, workplaces, and robot bodies. The long-term goal is a system in which active perception selects what to inspect, a grounded planner chooses and revises subgoals, motor policies execute at the timescale of physical control, and outcome verification updates memory and future decisions. This organization is a plausible route toward autonomy in bounded operational domains and, eventually, toward the broader generality sought. It is a research direction, not a capability already achieved.

1.2 Ladder of Embodied Autonomy

To establish a rigorous taxonomy for robotic capability progression, we formalize the six levels of embodied autonomy in Table 1. This hierarchy systematically delineates the shifting boundaries of the Operational Design Domain, control authority, and human-robot collaborative roles. Below, we dissect the state of maturity across these tiers, envision the frontier of L4/L5, and elucidate why *Agentic Embodied Intelligence* serves as the foundational paradigm shift required to transcend the current plateau.

L0 and L1. Levels L0 (Direct Teleoperation) and L1 (Assisted Primitives) represent the most mature baselines

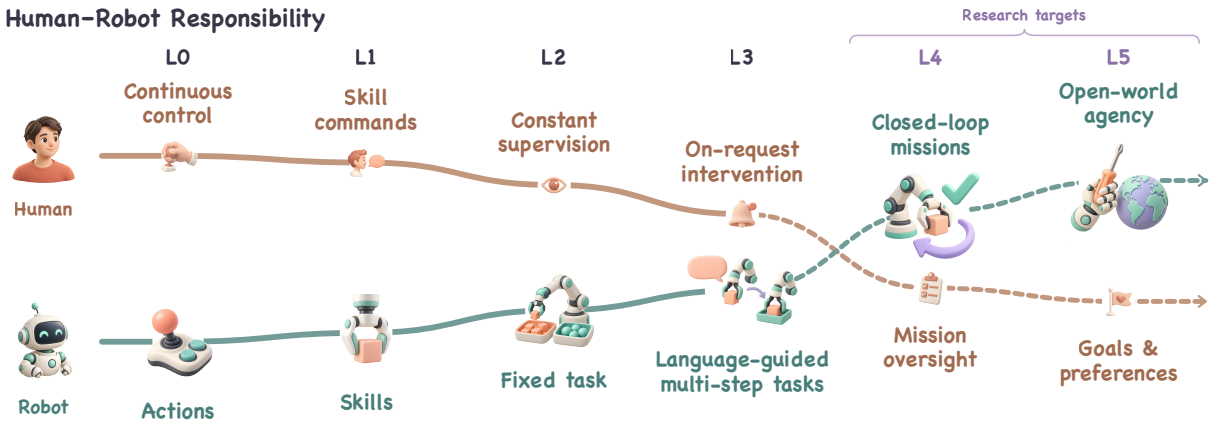


Figure 2 Conceptual human–robot responsibility ladder from L0 to L5. As robot capability advances from executing actions and skills to handling fixed tasks, language-guided multi-step tasks, closed-loop missions, and open-world agency, human involvement shifts from continuous control to skill-level commands, supervision, on-request intervention, mission oversight, and ultimately the specification of goals and preferences. The opposing trajectories illustrate this gradual transfer of responsibility, with L4 and L5 marked as research targets.

in robotics. L0 provides the sensory and kinematic backbone for safety-critical systems, notably surgical robotics [133] and large-scale teleoperation data acquisition rigs [442]. L1 forms the core of modern industrial automation, where robots execute isolated, closed-loop atomic primitives, such as trajectory replay via teach pendants [158], grasping primitives [257], single-query path planning [203]. At these stages, the physical machine exhibits zero cognitive independence, functioning purely as a deterministic mechanical extension of the human operator. Currently, L0 and L1 play a pivotal role as the indispensable data-generation engines fueling the entire embodied learning pipeline across both physical and simulated domains.

L2. In Level L2, autonomous capability is fundamentally bounded by a closed-set task manifold, where deep robotic learning achieved near-perfect dynamic fidelity for isolated, predefined workflows in structured environments. This phase is epitomized by classical end-to-end visuomotor policies [214], scalable closed-loop bin-picking systems [110, 182], and inductive spatial rearrangement architectures [426]. Furthermore, modern high-precision imitation frameworks [86, 442], alongside complex reinforcement learning specialists [32], demonstrated exquisite contact-rich motor dexterity. However, L2 controllers lack language comprehension and semantic reasoning; their policy manifolds overfit strictly to known geometries and controlled lighting. Any out-of-distribution (OOD) perturbation immediately triggers execution failure, necessitating continuous, on-site human supervision.

L3. The progression from L2 to L3 marks the watershed transition from task-specific specialist controllers to open-vocabulary foundation policies. The integration of multimodal foundation models catalyzed the leap to Level L3, wherein physical actions are conditioned on open-vocabulary natural language instructions. Represented by landmark Vision-Language-Action (VLA) models [60, 65, 66, 196] and World Action Models (WAM) [147, 195, 221, 283], L3 generalist policies leverage internet-scale pretraining to zero-shot generalize across novel object instances, semantic synonyms, and diverse spatial layouts. Driven by this promise, intense efforts [159, 227, 371] across both leading research institutions and industry frontier labs continue to heavily scale and advance the physical dexterity, high-frequency force control, and cross-embodiment robustness of foundation policy models. Indeed, a sufficiently capable, robust, and dexterous policy model remains an indispensable prerequisite and the physical bedrock upon which higher-tier L4 and L5 autonomy must be grounded. Despite broad semantic flexibility, L3 remains fundamentally conditional. Because monolithic VLAs operate as open-loop trajectory rollouts without explicit multi-step causal reasoning, active epistemic perception, or in-situ error diagnosis, they are intrinsically vulnerable to compounding covariate drift and physical deadlocks.

Vision for L4 and L5. The L4 target is a robot that can accept a mission, gather missing information, execute

and verify each step, recover from routine failures, and recognize when it must stop or raise an exception. Unlike L3, which remains an open-loop execution regime, L4 demands system-level self-closure. Level L5 represents the ultimate aspiration of embodied intelligence: open-world, unbounded general-purpose physical agency. Operating in arbitrary, unstructured environments (e.g., complex households, disaster relief zones, dynamic outdoor construction), an L5 system behaves as a fully competent physical partner. It requires zero specialized engineering oversight; non-expert end-users interact purely through abstract goals and subjective preferences. An L5 entity possesses human-like commonsense reasoning, dynamic tool fabrication capabilities, active epistemic disambiguation, and perpetual lifelong self-evolution while guaranteeing intrinsic physical safety. Recently, no evidence in the surveyed literature establishes sustained, unattended L4 operation across a broad, explicitly validated domain.

Why the agentic loop matters for L4 and L5. The transition from L2/L3 to L4/L5 changes the unit of autonomy from an action or demonstrated task to a sustained mission under uncertainty. A mission requires the robot to ground ambiguous goals, choose information-gathering actions, compose and dispatch skills, verify physical outcomes, recover or ask for help, and retain useful experience. These are precisely the functions of the *perception, planning, execution, reflection* loop developed in this survey. In this functional sense, agentic embodied intelligence is necessary for L4 and L5: a system must possess these closed-loop capabilities. Currently, with the development of coding agents [38, 281], it provides the cognitive syntax, execution rigor, and self-debugging mechanics that make physical closed-loop self-healing mathematically and operationally feasible [101, 435, 437]. While the architectural claim does not imply that an LLM alone is sufficient, where fast control, explicit safety limits, and validated handover remain essential, the successful experience in digital agents is of great referential value for construction of future embodied agentic systems.

1.3 Scope and Contribution

This survey studies agentic embodied intelligence (AEI) as a system-level approach to goal-directed physical behavior, in which models and robot policies operate within repeated cycles of perception, decision, action, and feedback. We organize the scope around the capabilities and interfaces needed to sustain this cycle, rather than around a particular model family or a requirement that every function be implemented as a separate module. We situates this view in the development of embodied systems and uses the L0–L5 autonomy ladder to distinguish established forms of control and task-bounded autonomy from the still-open goals of sustained L4 missions and L5 open-world agency. The survey's scope follows the path from intention to grounded action and back again, while asking what system conditions make that path reliable and deployable.

Section 2 establishes the two connected views that organize the survey: the system-level path from model through infrastructure and embodiment to the environment, and the model-level loop that links perception and grounding, reasoning and planning, action execution, and closed-loop evolution. Sections 3–6 then trace how this loop works in practice. Section 3 examines how semantic, spatial, and physical grounding, together with passive and active perception, establish and update the task-relevant state on which an agent can rely. Given that state, Section 4 studies how goals and constraints are formulated, decomposed, searched, and represented as plans; Section 5 follows those plans into conditioned actions, policy representations, action generation, and control. Since issuing an action does not establish that its intended effect occurred, Section 6 examines verification, recovery, reflection, memory, and adaptation, through which outcomes revise the current behavior and inform future interactions. These functions are coupled rather than a rigid pipeline: a planner may request further perception, execution may expose an invalid assumption, and feedback may return the agent to an earlier decision.

The remaining sections ask how this online loop can operate beyond a single model call or isolated task. Section 7 examines the computing infrastructure, runtimes, interfaces, and hardware that coordinate model inference with robot execution across device, edge, and cloud resources. Section 8 treats safety and trustworthiness as constraints on the entire lifecycle, from interpreting evidence and selecting plans to executing actions, updating memory, and escalating to human oversight. Interaction also produces experience: Section 9 follows that experience from data sources and curation through training and the embodied data flywheel, while Section 10 considers simulators, benchmarks, and evaluation protocols as controlled environments for testing capabilities, diagnosing failures, and measuring whether progress transfers to physical deployment. Building

on these connections, Section 11 identifies open challenges and research directions across the loop and its supporting system, and

This survey makes following contributions:

1. We present a system-level framework that connects the online path from model to infrastructure, embodiment, and environment with the feedback path that returns observations and outcomes. Data, evaluation, and safety are treated as interdependent parts of this system rather than as post hoc add-ons.
2. We use an L0–L5 autonomy ladder to relate robot capability to human responsibility and operational boundaries. This framing distinguishes the conditional autonomy of current systems from L4 mission-level autonomy and the open-world ambition of L5, and motivates the agentic closed loop as a necessary architectural condition for higher autonomy, while recognizing that it is not sufficient without validated safety and deployment evidence.
3. We organize the model core into four coupled functions: perception and grounding, cognitive reasoning and planning, action execution, and closed-loop evolution. This taxonomy emphasizes the interfaces that make grounded state usable for planning, plans executable by robot policies, and physical outcomes available for verification and revision.
4. We synthesize the enabling system layers that connect these functions to deployment, including device-edge-cloud computation, runtimes and hardware, system-level safety, and an embodied data and evaluation ecosystem. In particular, we connect data sources organized by scale and cost to agentic experience records and feedback-driven improvement, and relate benchmark design to the capabilities it can validly measure.

2 Conceptual Framework

2.1 System Organization

Agentic embodied intelligence is best understood as a system rather than as a language model connected directly to a robot. As shown in Figure 1, the model interprets goals, estimates task-relevant state, and proposes plans or actions; an execution infrastructure mediates these proposals through skill and tool interfaces, scheduling, and communication; the embodiment supplies sensors, actuators, and low-level control; and the environment returns observations and physical consequences. The resulting interaction is bidirectional: commands flow from model to infrastructure and robot, while observations, contact signals, and verified outcomes flow back to the model. This separation allows semantic reasoning to operate at a slower timescale than stabilization and contact control, while keeping both accountable to the same task state.

The online stack is complemented by two offline-facing subsystems. The data ecosystem collects demonstrations, robot trajectories, and interaction outcomes, then curates them for pretraining, adaptation, or policy improvement; large-scale cross-embodiment resources make this learning loop increasingly important . Evaluation provides repeatable tasks, perturbations, and measurements of success, recovery, efficiency, uncertainty, and safety events. Its results should diagnose coverage gaps and failures, guide targeted data collection, and inform model or system updates, rather than serve only as a final leaderboard score. Safety and trustworthiness cut across both offline and online operation: model-level uncertainty checks, runtime validation, control limits, monitoring, and human takeover jointly constrain what the system may do. In short, the model proposes; infrastructure mediates; the embodied controller acts; the world responds; and data, evaluation, and safety determine how the system learns and where its operation must stop.

2.2 Model Cognitive Loop

The model core comprises four interdependent functions, not four isolated stages. Perception and physical grounding transform multimodal observations and task context into robot-relevant entities, spatial relations, affordances, and a state estimate that preserves uncertainty. Cognitive reasoning and long-horizon planning use this grounded state to interpret the goal, decompose it into subgoals, reason about dependencies and constraints, and decide whether more information is needed before acting. Action execution translates a



Figure 3 caption

selected plan into a skill call, motion target, or motor policy and carries it out through the appropriate controllers. Closed-loop evolution compares expected and observed outcomes, updates memory and state beliefs, diagnoses failure, and chooses among continuing, replanning, retrying, gathering more evidence, or handing control to a person. The forward path and return path are summarized in Figure 3.

The interfaces between these functions are central to agency. Perception must provide more than a fluent scene description: it should expose grounded references, spatial frames, and uncertainty that planning can use to test preconditions. Planning, in turn, should issue executable and monitorable subgoals rather than an unstructured narrative. Execution must return observations and action status, including failures and partial completion, instead of reporting only that a command was issued. Finally, closed-loop evolution must treat physical evidence as authoritative: it updates the working belief and memory, then routes the result to perception when the scene is ambiguous, to planning when the current plan is invalid, or to execution when a bounded retry is justified. Prior work on feasibility-aware language planning and feedback-conditioned replanning provides early examples of these interfaces; learned visuomotor and diffusion policies provide complementary action mechanisms but still require outcome monitoring.

This organization is a functional decomposition, not a claim that every system must contain four separately trained modules. A unified VLA model may internalize parts of perception, reasoning, and action, while external infrastructure supplies memory, monitoring, and safety checks. What makes the system agentic is that the functions remain causally connected: physical observations can change beliefs, beliefs can change plans, plans can dispatch bounded actions, and measured outcomes can change what happens next. Without this return path, an apparent multi-step capability is only open-loop action generation; with it, the agent can detect divergence between intended and actual state and adapt its behavior accordingly.

3 Perception and Grounding

An embodied agent needs task-relevant evidence about its environment to plan and act. To "put the mug in the dishwasher", it must identify the intended mug, locate it relative to the robot, and determine whether its contents and the dishwasher's current state permit the intended interaction. Perception provides the semantic, geometric, and physical information needed to check action preconditions, guide execution, and assess outcomes. Yet this information is necessarily incomplete: occlusion hides relevant objects and states, sensor noise limits estimation accuracy, and both the agent's actions and external events change the environment. An agentic system must therefore maintain its estimates throughout the task and recognize when additional evidence is needed before proceeding.

We organize these capabilities through a *belief-state perspective*. In the partially observable Markov decision process formulation, a belief state is a probability distribution over possible world states conditioned on the history of observations and actions. Here, this perspective provides a conceptual framework for organizing task-relevant estimates and their uncertainty; it does not assume that every reviewed system explicitly maintains such a distribution. We distinguish three coupled dimensions of the estimated state. *Semantic grounding* connects task descriptions to particular objects and functional parts, resolving what the instruction refers to. *Spatial grounding* establishes their geometry, poses, and spatial relations in reference frames usable by the robot. *Physical grounding* estimates the properties, contact states, and mechanical constraints that govern how objects respond to interaction and whether an intended action is feasible. These dimensions describe complementary information requirements rather than sequential processing stages: identifying a handle, locating it, and determining how it can move jointly constrain the same interaction.

Maintaining this information requires both incorporating available evidence and acquiring evidence that is missing. *Passive belief maintenance* associates incoming observations with previously observed entities, revises changed states, and determines whether earlier evidence remains relevant and valid. When available observations cannot resolve an uncertainty that matters to the next decision, *active information acquisition* selects an additional viewpoint, a physical probe, or a clarification question according to its expected decision value and its cost in time, effort, risk, or human attention. These mechanisms build on established foundations: probabilistic robotics [347] provides tools for estimation under uncertainty; active perception [50] and active vision [34] treat the acquisition of observations as a process that can be controlled; and interactive perception [62] exploits action-induced sensory changes to reveal otherwise ambiguous properties and states. Foundation models [166, 297] extend the representational resources available to this loop. The agentic requirement is to use these representations within an ongoing process of estimating the world, identifying consequential information gaps, and gathering evidence to support subsequent decisions.

Accordingly, this section is organized around the content of task-relevant state estimates and the mechanisms that keep them useful. Sections 3.1, 3.2, and 3.3 review semantic, spatial, and physical grounding, respectively. Section 3.4 then examines passive belief maintenance and active and interactive perception across these three dimensions. Together, these capabilities supply the state estimates, feasibility evidence, and outcome observations used by downstream planning, action execution, and closed-loop reflection.

3.1 Semantic Grounding

Semantic grounding establishes which objects or parts a task expression refers to in the observed scene and relevant context. The same cup may be identified as "the red cup," "the cup next to the kettle," or "something to drink from." These expressions rely on different evidence: the target's category and attributes, its relations and context, or its function and intended use. We use this distinction to organize the literature. The three forms of reference often coexist, and each can identify either a whole object or a particular part.

3.1.1 Category- and attribute-based grounding

Category names and attributes provide the most direct connection between language and visual evidence. Vision–language pretraining, particularly CLIP [297], made recognition possible through text queries beyond a fixed label set. Grounding requires carrying this semantic alignment into individual image regions. Early open-vocabulary detectors pursued this through distillation and detection fine-tuning, respectively, in ViLD [129]

Table 2 Taxonomy of agentic perception.

Component	Approaches	Benchmarks
What the belief contains		
Semantic grounding (§3.1) *Which objects and parts does the task refer to?* Evidence: images, language	Open-vocabulary detection, segmentation [75, 242] Generative and prompted grounding [99, 208, 396] Parts and dense correspondence [180, 240]	LVIS [131], POPE [230]
Spatial grounding (§3.2) *Where are they, and in which frame?* Evidence: depth, multi-view, kinematics	Feed-forward geometry, object pose [357, 375, 403] Spatial VLMs and cognitive maps [78, 364, 377] Action-oriented targets [153, 417] Scene graphs [128, 256]	BLINK [117], VSI-Bench [397], MMSI-Bench [402], MV-RoboBench [114], ScanRefer [79]
Physical grounding (§3.3) *What state are they in, and how will they respond?* Evidence: touch, force, sound, vision	Physically grounded VLMs and priors [119, 361] Tactile and tactile-language models [138, 414] Articulation models [176, 270] Learned dynamics and world models [44, 232]	PhysBench [89], QuantiPhy [292], STATUS Bench [349], Physion [56]
How the belief is updated		
Passive perception (§3.4.1) *What remains true as observations arrive?* Evidence: observations, odometry	Tracking with memory and touch [301, 337, 375] Consistent metric-semantic mapping [71, 73, 157] Change and staleness [212, 310] Spatio-temporal memory [41, 405]	OpenEQA [258], RoboMME [94]
Active perception (§3.4.2) *What is missing, and which action reveals it?* Evidence: viewpoints, contacts, answers	View planning and exploration [160, 411, 452] Calibrated stopping [304] Interactive probing [140, 173] Asking [306] Outcome verification [106, 449]	EQA [97], AbstainEQA [381], RoboFail [247], VLA-FixBench [392]

and OWL-ViT [268]. Joint training on detection and phrase-grounding data subsequently brought richer descriptions into localization, as demonstrated by GLIP [222] and Grounding DINO [242].

The difficulty shifts from recognizing individual concepts to preserving their composition. A detector may recognize cups, bowls, and colors yet confuse "the red cup" with a nearby red bowl. ARO [420] exposes such failures in attribute binding and relational understanding, showing why broad vocabulary coverage alone is insufficient. The same requirement applies at finer granularity: "the drawer handle" must identify the correct part of the correct object. VLPart [335] and PartSLIP [240] extend semantic recognition to parts in images and point clouds, respectively, providing the referents whose geometry is further resolved by spatial grounding.

3.1.2 Relational and contextual grounding

An object's appearance may not distinguish it from nearby alternatives. In "the cup next to the kettle," identity depends on another object and the relation between them. This motivates interpreting the full expression together with the scene, rather than matching isolated category words. MDETR [183] brings language into the detection process, while Kosmos-2 [286] connects generated text to localized regions. An alternative is to make candidate regions directly addressable: Set-of-Mark prompting [396] adds visual identifiers that a multimodal model can select when interpreting the instruction. Regardless of the interface, identifying the cup and kettle separately does not establish that the selected pair satisfies the requested relation.

Some references cannot be resolved from the current image alone. "My cup" requires prior knowledge of a particular instance; "the cup I used earlier" additionally invokes interaction history. Personalization supplies this missing context through learned concepts or reference images. MyVLM [33] incorporates user-specific concepts into visual understanding, and Visual Attentive Prompting (VAP) [211] connects reference-image matching to robotic manipulation. For references grounded in earlier events, the agent must also preserve object identity across observations. This dependence links semantic grounding to belief maintenance (Section 3.4.1): recalling an earlier object is useful only if that evidence can be associated with the current candidate.

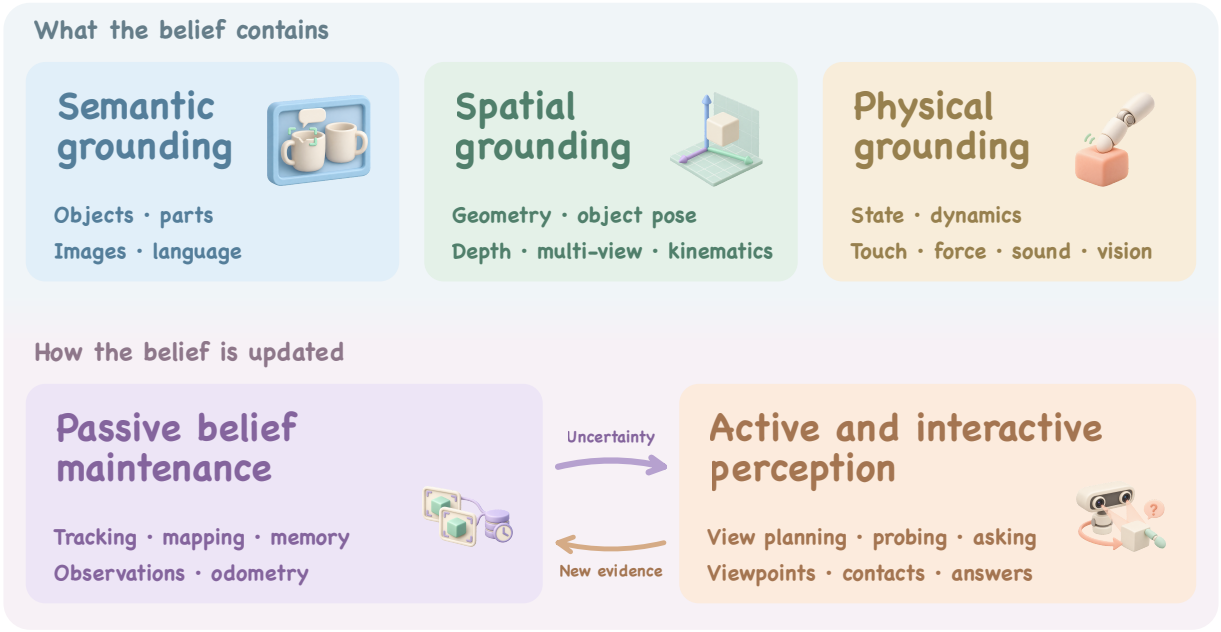


Figure 4 A belief-centered taxonomy of embodied perception. The upper row organizes task-relevant belief content into semantic grounding, spatial grounding, and physical grounding. The lower row shows how this belief is maintained and refined: passive belief maintenance integrates observations, odometry, mapping, and memory, while active and interactive perception selects viewpoints, probes, or questions when uncertainty matters to a decision. New evidence from these information-seeking actions returns to belief maintenance, closing the perception loop.

3.1.3 Function- and intent-based grounding.

A task can specify what an object is needed for without naming it. "Find something to cut the tape with" requires connecting an intended use to a suitable entity in the scene. TOIST [224] formulates this as task-oriented instance segmentation, where the action description guides object selection. LISA [208] extends segmentation to implicit queries requiring reasoning and world knowledge. Together, these directions move beyond explicit object descriptions toward identifying targets from the needs expressed by the user.

Functional evidence can also come from observed interaction. Robo-ABC [180] transfers contact points from human-video examples to unfamiliar objects through semantic correspondence, connecting visual regions to how objects are used. Such associations help identify a relevant tool or part, but leave physical questions unresolved. Scissors may fit the purpose of cutting while being unsuitable for the material or impossible for the robot to operate. These conditions belong to physical grounding (Section 3.3).

A grounded expression need not yield a single confident answer. Several objects may satisfy it, or no visible object may match. GRES [238] makes these cases explicit in referring-expression segmentation, and Rex-Thinker [175] examines candidates before accepting or rejecting a match. For an embodied agent, the result should preserve the supported candidates and any unresolved ambiguity. Failure to identify a target in one view also does not establish its absence from the environment. When this distinction matters to the next action, grounding must remain open to additional observation or clarification (Section 3.4.2).

3.2 Spatial Grounding

Spatial grounding connects observations and task referents to the geometry of the environment. It involves both *geometric estimation*, which recovers shape, scale, position, and orientation, and *spatial reasoning*, which interprets relations among entities under a specified perspective. Estimating the geometry of a cup and a drawer, for example, provides evidence for judging whether the cup is inside the drawer. These capabilities are

coupled: geometric estimates support relational judgments, while task descriptions determine which spatial quantities and relations need to be resolved.

3.2.1 Geometric Estimation

Depth and scale establish the geometric basis for locating surfaces and measuring spatial extent. Large-scale visual pretraining has broadened the applicability of monocular depth estimation, as demonstrated by Depth Anything [399] and Depth Anything V2 [400]. However, relative depth alone does not determine physical dimensions. Metric3D v2 [145] and UniDepth [288] explicitly target metric recovery, while stereo methods such as FoundationStereo [374] exploit correspondence between paired views. These approaches obtain scale from different combinations of learned priors and camera geometry. Their outputs therefore carry different assumptions: monocular metric predictions remain subject to generalization error, and converting stereo disparity to physical depth depends on calibration.

Multiple views provide additional constraints on surfaces that are only partially visible in any single image. Combining them requires establishing correspondences, recovering camera poses, and aligning observations into a common scene. DUSt3R [365], MASt3R [213], and VGGT [357] increasingly couple these tasks through learned point maps, matching features, and camera predictions. MapAnything [188] further accommodates optional geometric inputs, including calibration, poses, and partial depth. The resulting reconstructions provide a spatial basis for integrating semantic evidence across views. VLMaps [149] and ConceptFusion [166] attach visual–language features to mapped locations, while LERF [190] and LangSplat [294] support dense language queries through radiance-field and Gaussian representations. Such representations allow the agent to locate task-relevant entities beyond the current image; keeping them consistent as the environment changes is addressed in Section 3.4.1.

Within this scene, object pose estimation establishes the position and orientation of individual entities. The main distinction is how much object knowledge is available: PoseCNN [385] addresses known instances, NOCS [356] generalizes across instances within a category, and MegaPose [205] and FoundationPose [375] use test-time references to estimate poses of novel objects. When a reference model is unavailable, shape can be reconstructed alongside tracking, as in BundleSDF [373]. Occlusion and symmetry can nevertheless leave multiple poses consistent with the observations. Moreover, a pose estimated relative to the camera must be related to the robot through calibration and, where appropriate, forward kinematics. This alignment connects an observed object or part to the coordinates in which the robot plans and controls motion.

3.2.2 Spatial Reasoning

Spatial reasoning determines what spatial evidence implies: which object is nearer, whether one region contains another, or how several observations fit into a common layout. These judgments require more than recognizing the participating objects. What'sUp [184] exposes difficulties with basic spatial relations, BLINK [117] examines capabilities including depth comparison and visual correspondence, and VSI-Bench [397] tests spatial understanding from video. Across these settings, the challenge grows from interpreting a visible arrangement to integrating evidence distributed across viewpoints and time.

Perspective introduces a further source of ambiguity. "Left of the cup" may refer to the observer, the robot, or an object-centered frame; a viewpoint change can alter the correct description without changing the scene. Situated reasoning therefore requires establishing where the observer is and how it is oriented. SQA3D [253] makes this situation part of the question, while SIG3D [260] explicitly estimates it before answering. RoboSpatial [332] extends robot-oriented spatial supervision to egocentric, object-centric, and world-centric distinctions. Together, these settings emphasize that spatial relations must remain consistent with both the observed geometry and the perspective specified by the task.

Addressing these difficulties requires training signals and inputs that preserve spatial information. SpatialVLM [78] and SpatialRGPT [84] strengthen supervision on directions, distances, and regional relations through geometrically grounded data. Depth-aware encoding and three-dimensional features, used in SpatialRGPT [84] and 3D-LLM [139], also provide evidence that may be difficult to infer from appearance alone. Where answers can be checked against image structure or depth, reinforcement learning offers an

additional training signal: Spatial-SSRL [246] constructs self-supervised tasks with verifiable outcomes. The objective must still distinguish perceptual accuracy from elaborate reasoning. SpatialTree [386] finds that indiscriminately encouraging extended deliberation can improve some reasoning abilities while degrading low-level perception.

For questions involving several entities or views, explicit representations help organize the evidence used in reasoning. Scene graphs in Hydra [157] and ConceptGraphs [128] expose objects, places, and their relations as queryable structures. Cognitive maps serve a similar purpose during inference: experiments with MindCube [364] support constructing a spatial model before answering multi-view questions. Spatial computations can also be delegated to executable programs, as in VADAR [266], which combines generated procedures with visual tools. These approaches make intermediate assumptions accessible for inspection, although errors in detection, association, or geometric estimation can still propagate into the final judgment.

Geometric estimates and spatial judgments reach planning and execution through targets and constraints. Language-conditioned points in RoboPoint [417], semantic orientations in SoFar [293], three-dimensional value maps in VoxPoser [154], and relational keypoint constraints in ReKep [153] provide different ways to express the spatial requirements of a task. Their usefulness depends on the precision required by the intended interaction: an estimate adequate for approaching a table may be insufficient for grasping a narrow handle. The perception interface should therefore preserve the coordinate frame, supporting evidence, and uncertainty of its outputs, so that downstream modules can determine whether to proceed or acquire further observations.

3.3 Physical Grounding

Physical grounding connects task-relevant objects and candidate interactions to physical properties, states, and constraints, providing evidence about their likely effects and feasibility. Lifting a mug, for example, requires more than locating its handle: the load and contact conditions affect whether lifting will cause slipping and whether the robot can maintain a stable grasp. We organize this process around three questions: what physical conditions hold, what consequences an interaction would produce, and whether those consequences satisfy the task requirements within the robot's capabilities (Table **??**). These questions form a conceptual progression; answering them does not require a fixed computational pipeline.

3.3.1 Estimating Physical Conditions

Physical conditions include relatively persistent properties, such as mass and stiffness, alongside changing states, such as loading and contact. Their estimation depends on what the observations reveal. Appearance supports prior judgments about material and other physical attributes, aided by physical-concept supervision from the PhysObjects dataset [119]. Observed motion constrains otherwise hidden properties by exposing their effects; simulation-based inference in Galileo [379] connects these effects to candidate physical parameters. Interaction also reveals mechanical constraints: changes before and after manipulation can identify movable parts and articulation, as in Ditto [176].

Touch adds local evidence that vision may miss. Deformation observed by GelSight [418] supports estimates of contact geometry, force, and slip, while tactile-language representations in Octopi [414] support property descriptions and reasoning. Across these evidence sources, estimates can be refined as interaction unfolds. Phys2Real [361] combines visual-language parameter priors with online estimates and their uncertainty. The resulting knowledge may therefore take the form of an attribute judgment, a parameter estimate, or a contact-state observation; each constrains different aspects of the next interaction.

3.3.2 Predicting Interaction Outcomes

Given current conditions, physical prediction asks how an interaction will change the object or scene. Explicit models organize this change around interacting entities: Interaction Networks [55] and DPI-Net [232] use object relations and particle interactions to model dynamics, while material-conditioned prediction in AdaptiGraph [433] accommodates variation in physical properties. Such representations expose the entities and interactions used for prediction, but depend on suitable state estimates. Latent models instead predict

future representations without requiring explicit recovery of every physical quantity. V-JEPA 2-AC [44], the action-conditioned extension of V-JEPA 2, follows this route for robot interaction.

The prediction question also depends on what is allowed to change. Forecasting motion under observed conditions differs from asking how an event would unfold under an altered initial configuration. CoPhy [53] addresses the latter by inferring latent physical factors from an observed event and reusing them under modified conditions. For embodied tasks, the relevant prediction is the anticipated change in object state or contact, including its dependence on the proposed action and uncertain physical conditions. Whether that change achieves the goal is a further judgment.

3.3.3 Assessing Interaction Feasibility.

Feasibility assessment relates an expected effect to the task requirements and the robot's capabilities. An action may move an object yet fail to achieve the required displacement or preserve a stable grasp. Affordances capture this dependence through relations among objects, actions, and effects [271]. Their scope ranges from local interaction opportunities to the success of an available skill. Where2Act [270] estimates actionable regions and local motion proposals for articulated objects; action-conditioned visuotactile grasp models [72] assess candidate grasp adjustments using current visual and contact evidence. At the skill level, SayCan [27] combines learned skill-value estimates with language-based task relevance.

These estimates can be obtained from predicted physical outcomes or learned directly from interaction experience. Direct assessment need not reconstruct a complete physical model, but its meaning remains tied to the evaluated action, intended effect, and available controller. In particular, a skill-value estimate reflects success under a particular policy rather than certifying physical possibility. Physical grounding supplies this evidence for planning; selecting and sequencing actions belongs to the downstream decision process. When the evidence is insufficient, active perception can seek observations that resolve the relevant uncertainty.

3.4 From Passive Perception to Active Perception

Grounding a scene once does not keep it grounded as the robot and the world move. Belief updating must both incorporate new evidence and identify when the available evidence is insufficient. *Passive belief maintenance* updates the estimate from incoming observations. *Active and interactive perception* selects actions to obtain missing evidence. This distinction concerns why an action is chosen: observations acquired during ordinary navigation or manipulation can support passive maintenance, even though the robot is moving. The two processes form a loop, since maintenance exposes uncertainty and information-gathering actions produce evidence to incorporate.

3.4.1 Passive Belief Maintenance

Incoming evidence raises three questions: which stored entities it concerns, whether it invalidates an earlier estimate, and which observations remain useful for the current decision. These correspond to maintaining correspondence, revising outdated state, and retrieving relevant evidence. Together they determine whether a persistent representation remains a useful estimate of the world rather than a record of what was once observed.

Correspondence and consistency. A change in appearance need not indicate a change in identity, and a change in image position need not indicate object motion. The robot must separate viewpoint changes, estimation errors, and actual scene changes before combining observations. At the object level, temporal masks, jointly tracked points, and pose estimates maintain correspondence across frames [187, 301, 373, 375]. During manipulation, visual occlusion makes contact evidence especially useful: combining vision and touch can constrain object shape and pose when the gripper hides much of the surface [218, 337]. Such tracking supports continuity through missing observations, but does not make the hidden state directly observed. At the scene level, repeated visits constrain accumulated localization error through loop closure and global optimization [73]. Corrections must propagate to the objects and relations expressed in the map, including the layers of a scene graph [157]. Semantic correspondence adds a further difficulty because different views of

the same object yield different features and descriptions. Multi-view association consolidates them into object-level representations [128], while probabilistic semantic updates retain uncertainty in the fused estimate [71]. Geometric agreement, semantic agreement, and confidence therefore play different roles; averaging repeated features alone does not establish that an object has been identified reliably.

Change detection and temporal validity. A geometrically consistent map can still be wrong about the present. A cup may have moved, a drawer may have closed, or an object may have been removed since the last visit. Unlike drift, these changes should revise the represented world rather than be absorbed as errors in the robot's trajectory. Object-level mapping separates movable entities from the surrounding geometry [311], and spatio-temporal estimation treats short-term motion and longer-term changes at different time scales [310]. Dynamic scene graphs and point-cloud memories likewise update object states as new evidence arrives [241, 393]. Unobserved change is harder: an old observation cannot establish that its contents remain true. Temporal validity depends on both the time since observation and how frequently the relevant state changes. Volatility-aware memory uses this distinction to revise or retain stored hypotheses [212]. Absence also requires care, since a missed detection may reflect occlusion rather than removal. Maintenance should preserve the difference between an observed change, a predicted change, and an unresolved possibility. Otherwise, a planner can mistake an old location or an unverified absence for a current fact.

Task-conditioned evidence retrieval. Keeping every observation is neither necessary nor sufficient for a useful belief. A grasp requires recent pose and contact evidence, whereas a question about the last sighting of an object requires historical evidence. Spatial and temporal databases support queries over previous observations [41]; compact multi-view snapshots retain visual context for exploration and reasoning [405]; and hierarchical retrieval limits the search to relevant spatial scales [387]. Learned persistent states provide another form of compression [362], but make the connection between an answer and its supporting observation less explicit. The main trade-off is between compactness and the ability to recover evidence that later becomes relevant. A retrieved observation should retain its location, time, and status as current evidence or historical context. Successful recall of a previous scene does not establish the present scene state. Here memory supports the current estimate and its evidential basis, where reusable experience and skill learning are helpful.

3.4.2 Active and Interactive Perception

Maintenance cannot resolve ambiguities that the available observations do not distinguish. Active perception treats sensing as a control problem [34, 50, 51], and interactive perception uses physical action to generate informative sensory responses [62]. Three mechanisms supply additional evidence: changing the observation, changing the scene, and asking a person. Verification and stopping govern their use across the task. The relevant objective is to resolve uncertainty that can affect a pending decision, rather than to maximize coverage or collect experience for future learning.

Viewpoint selection and targeted search. Some uncertainty comes from limited visibility: the target may be outside the field of view, behind another object, or seen from an uninformative angle. The action should therefore be chosen for the evidence expected at its destination. Next-best-view methods score candidate views by expected visibility or reconstruction gain [92, 160]; frontier exploration extends this principle to the boundary of known space [391]. Semantic maps and vision–language scores prioritize regions likely to contain a target [76, 411], while learned viewing policies adapt observations to the downstream task [90, 168]. These approaches differ in what they optimize. Improving scene coverage, finding an object category, and resolving a grasp-critical occlusion require different views. Semantic relevance can guide search without providing calibrated information gain, and a complete map may contain little evidence about the particular attribute in question. View selection should consequently reflect the unresolved variable and the cost of reaching a useful observation, rather than treating all unexplored regions as equally informative.

Physical probing and scene intervention. A better view cannot reveal every physical variable. Contact can expose resistance or compliance, and opening or moving an object can reveal previously inaccessible contents. Interactive perception couples the chosen action with the resulting sensory change, so that the response is interpreted conditional on how it was produced [62]. Motion before and after interaction constrains articulation [176], while metric-semantic belief-space planning jointly considers sensing and manipulation under occlusion [265]. Action-conditioned scene representations record which interventions expose which

Table 3 Questions and representative approaches for belief maintenance and information acquisition. Works can support multiple functions; verification and stopping apply across acquisition mechanisms.

Aspects	Questions	Approaches
Passive belief maintenance		
Correspondence and consistency	How does new evidence connect to the stored scene?	Hydra [157]
Change detection and temporal validity	Which stored states must be revised?	Khronos [310]
Task-conditioned evidence retrieval	Which observations support the current query?	3D-Mem [405]
Active and interactive perception		
Viewpoint selection and targeted search	Where should the robot observe next?	VLFM [411]
Physical probing and scene intervention	Which interaction can reveal the hidden state?	RoboEXP [173]
Human clarification	Which ambiguity should be resolved by asking?	KnowNo [306]
Verification and stopping	Has the expected effect occurred? Is the evidence sufficient to answer?	Code-as-Monitor [449] Explore-until-Confident [304]

entities [173]. These approaches connect information gathering to physical grounding while also updating the scene that subsequent actions will encounter. An informative intervention can simultaneously advance the task and change the state being estimated. Its value therefore depends on both what it reveals and what it leaves possible afterward. The probe should distinguish plausible hypotheses with acceptable disruption and risk. Learning predictive models from poking provides a foundation for interpreting action effects [25], but collecting such training experience does not by itself specify which probe is useful for a current decision.

Human clarification. Other ambiguities concern intended meaning rather than an unobserved physical state. If two cups match an instruction, additional camera views may not reveal which one the user intended. Interactive grounding asks questions that distinguish candidate referents [322], and belief-based decision making compares asking with acting on the current estimate [431]. Calibrated prediction sets provide a way to request help when multiple action options remain plausible [306]. The guarantees assume representative calibration data, correctly grounded inputs, reliable human help, and successful execution of selected actions. They therefore do not certify the perception and control modules themselves. A useful question targets the unresolved choice and offers alternatives grounded in the scene. Asking about a visible distinguishing attribute can resolve reference, whereas asking the user to restate the entire task may add little information. Human attention is part of the cost, so clarification is justified when the response can change the decision and is preferable to further sensing or probing.

Information gathering requires a criterion for when the evidence is sufficient. Calibrating answer confidence can support stopping in embodied question answering [304], but confidence must be related to the required decision and its error cost. A model's confident answer alone does not establish that a grasp is stable or a requested placement has succeeded. Verification instead checks the expected effects of the action: scene or success feedback supports replanning [156], while explicit spatial and temporal constraints enable targeted monitoring during execution [449]. Verification is a purpose shared by the three information sources, rather than a separate sensing modality. Monitoring incoming observations is passive; selecting an extra view, contact test, or question to check an outcome is active. The evidence and its timing must match the possible failure: a final image may show placement without establishing stability, and a late check may miss an earlier opportunity to recover. A useful result distinguishes verified success, observed failure, and insufficient evidence, allowing the agent to continue, seek information, or hand the discrepancy to control and reflection. Table 3 summarizes these functions and representative approaches.

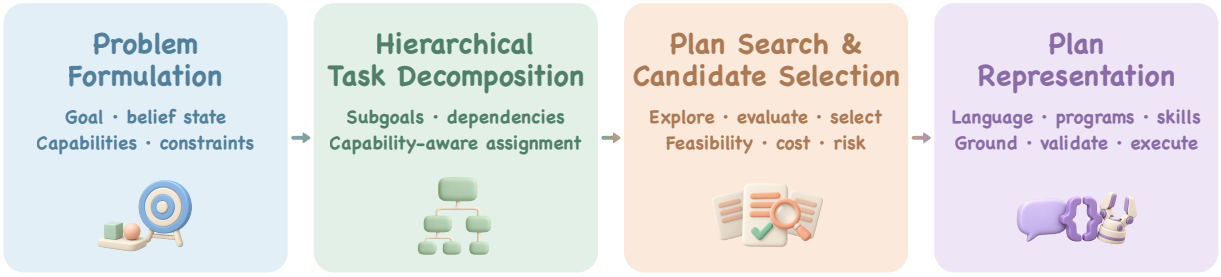


Figure 5 Conceptual organization of cognitive reasoning and long-horizon embodied planning. Problem formulation establishes the goal, belief state, available capabilities, and constraints. Hierarchical decomposition constructs candidate subgoals and their dependencies; search compares alternative branches and anticipated outcomes; plan representation grounds the selected plan in skill interfaces, programs, or constraints for validation and execution. Solid arrows indicate the main information flow, while dashed arrows indicate revision and replanning driven by constraint checks or observed outcomes. These functions may be interleaved in practical systems.

4 Reasoning and Planning

Embodied reasoning and planning provide the deliberative foundation for agentic embodied AI, enabling agents to organize their interactions with the physical world in pursuit of high-level goals. They connect grounded understanding of the environment with purposeful action by determining what should be achieved, how a task should be structured, and which course of action should be pursued. This chapter reviews how embodied agents formulate planning problems, construct and evaluate plans, and express their decisions in forms that support physical execution.

As illustrated in Figure 5, this chapter is organized around four interconnected components. Section 4.1, *Problem and Requirements*, establishes the goals, state information, available capabilities, and constraints that define long-horizon embodied planning. Building on this formulation, Section 4.2, *Hierarchical Task Decomposition*, examines how high-level goals are structured into subgoals and their dependencies. Section 4.3, *Plan Search and Candidate Selection*, reviews how planners explore alternative courses of action, evaluate their anticipated consequences, and select among candidates based on task progress, feasibility, cost, and risk. Section 4.4, *Plan Representation*, discusses how planning decisions are expressed through language, programs, constraints, or callable skills to support grounding, validation, and execution. Together, these components connect task intent to executable behavior. Their relationships extend beyond a fixed sequence: representations and skill interfaces shape decomposition and search, while feasibility checks and execution feedback inform candidate selection and revisions to the plan.

4.1 Problem Formulation

Embodied planning seeks to determine how a robot can achieve a task goal through interactions with its environment. A high-level instruction specifies the intended outcome but may leave the intermediate steps, their ordering, and their physical realization unspecified. The planning problem is therefore to construct a course of action that connects the robot's current situation to the desired outcome using its available capabilities. This chapter focuses on long-horizon tasks spanning multiple actions or skills, in which intermediate outcomes shape subsequent decisions and must be considered in relation to the overall goal.

Let s_t denote the world state at time t, including the robot configuration, object states, and task-relevant relations. The robot receives observations o_t, from which it obtains a state estimate or maintains a belief $b_t(s)$ over possible states. Given a high-level instruction g and an available skill set $\mathcal{S}$, a sequential plan for a skill-based system can be abstracted as

$$\Pi = \big[(g_1, \sigma_1, \theta_1), \ldots, (g_n, \sigma_n, \theta_n)\big], \tag{1}$$

where g_i specifies an intermediate goal, $\sigma_i \in \mathcal{S}$ is a selected skill or policy, and θ_i contains its arguments and grounding information. The plan thus specifies both the intermediate outcomes to pursue and the capabilities

through which they are to be achieved. The sequence provides a basic abstraction; richer structures can express partial ordering, shared resources, or conditional branches. Incomplete observations and uncertain skill outcomes may also require the remaining plan to be revised during execution.

The central planning question of agentic embodied intelligence is how these components collectively lead to the intended task outcome. Decomposing an instruction into locally meaningful subgoals does not by itself establish that their completion will satisfy the overall goal: necessary intermediate conditions may be omitted, dependencies may be misordered, or individual choices may conflict with later objectives. A plan must therefore connect the intended outcome to a coordinated progression of intermediate states. In embodied settings, this progression depends on the robot's available capabilities and on how its actions change the environment. Skill selection and parameter choices determine which transitions are possible, while the resulting states determine which subsequent actions remain available. Task-and-motion planning (TAMP) [121] formalizes an important aspect of this relationship by coupling discrete task decisions with continuous physical feasibility. Constructing a complete plan therefore requires reasoning jointly about the contribution of intermediate goals, their dependencies, and the actions through which they can be realized.

The resulting problem is to construct a coherent, executable course of action that satisfies the task goal and its constraints, while accounting for factors such as execution time, resource use, and risk when comparing alternatives. The following sections examine three complementary aspects of this problem: hierarchical task decomposition establishes intermediate goals and their dependencies; plan search and candidate selection explore and compare alternative ways to achieve them; and plan representation expresses the resulting decisions in forms that support grounding, validation, and execution.

4.2 Hierarchical Task Decomposition

Hierarchical task decomposition connects an overall objective to intermediate units of reasoning and execution. Its organization reflects two complementary considerations: the structure of the task itself and the capabilities of the components that carry it out. The first determines which intermediate outcomes contribute to completion and how they depend on one another; the second determines how much work can be assigned to an executor without further subdivision. We examine these considerations first in general LLM-based systems and then in embodied systems, where task structure and execution capability are additionally connected through the physical state of the world.

Task structure in general LLM systems. From the perspective of task structure, decomposition reduces the scope of individual reasoning problems while retaining their contribution to the overall objective. Its value lies in exposing intermediate results that later steps can use and in making the relationships between those results explicit. Sequential subproblems are appropriate when each answer supports the next, whereas nested subtasks allow an intermediate objective to be elaborated at a finer level. Least-to-Most Prompting [448] and Plan-and-Solve [359] exemplify the broader principle of organizing intermediate reasoning before completing the overall solution. The central issue is whether the chosen structure preserves the dependencies needed for completion without introducing unnecessary intermediate steps.

Capability-awared decomposition in general LLM systems. From the perspective of execution, a task has no universally appropriate decomposition depth. A subproblem that one component can solve directly may require several operations for another, and a useful division of work should reflect this difference. Specialization therefore affects both where subtask boundaries are placed and how their outputs are combined. Decomposed Prompting [192] and HuggingGPT [318] illustrate decomposition around specialized computational components, while ADaPT [291] illustrates refinement according to executor outcomes. Modular delegation and adaptive refinement address different aspects of executor compatibility: the former allocates responsibility, while the latter adjusts the amount of work assigned as a unit. They can coexist within the same hierarchy. The transition to embodied tasks changes what these intermediate units must accomplish. A subgoal may require establishing an object relation, preparing an interaction condition, or moving the robot into a suitable configuration. Its completion changes the conditions under which the rest of the task will be attempted. Task decomposition must consequently relate the semantic organization of the instruction to a progression of physical states, while matching each part of that progression to available execution capabilities. The same two considerations now operate together within a changing environment.

Task structure in embodied systems. In embodied systems, task decomposition must connect the intended outcome to intermediate objectives that reflect the current environment. The same instruction may require different subtasks depending on object locations, existing relations, and the conditions under which actions can be performed. These conditions determine which preparatory steps are necessary and how their completion supports subsequent objectives. A task hierarchy therefore organizes more than the semantic content of an instruction: it relates the overall goal to progressively more specific objectives and the dependencies among them. LLM-Planner [333] and MLDT [382] illustrate grounded task-step generation and explicit hierarchical decomposition, while SayPlan [299] and DELTA [244] incorporate structured scene information into task planning. Across these approaches, the central concern is how to organize intermediate objectives so that they collectively account for the steps required to achieve the goal in the given environment.

Executor-conditioned decomposition in embodied systems. At the execution level, the question is where high-level decomposition should stop and responsibility should pass to a skill or policy. Broad subgoals allow a capable executor to resolve local details, but may leave essential decisions unspecified for a weaker one. Excessively fine decomposition can instead duplicate decisions already handled by the policy and introduce additional calls and coordination overhead. Named-skill composition in SayCan [28], hierarchical model designs such as HiP [31] and Hi Robot [320], and multi-level reasoning in Gemini Robotics 1.5 [345] illustrate different ways to connect reasoning with action. The appropriate granularity therefore depends on the execution interface and the range of decisions delegated to the underlying controller. Executor compatibility also extends across subtask boundaries. Two subtasks may each have a suitable policy, yet their composition can fail if the first leaves conditions outside the second policy's operating range. Decomposition and assignment are consequently coupled: changing the executor can change the required subgoal parameters, preparatory steps, or point of handoff. RoboHarness [151] makes this connection explicit through capability-aware decomposition and heterogeneous policy orchestration. This perspective treats an intermediate boundary as both the completion of one unit of work and the establishment of conditions for the next. With multiple robots, the same relationship extends to responsibility allocation and coordination. Subtasks that are semantically independent may still compete for workspace or resources, while others require synchronized contributions from different robots. The task structure must therefore accommodate both goal dependencies and the conditions of joint execution. Collaborative and hierarchical systems such as RoCo [261], EmboTeam [427], and IMR-LLM [334] provide examples of this broader setting. Their relevance here is the coupling of task division, executor assignment, and coordination, which extends the executor-matching problem from individual capabilities to team organization.

Task structure and execution capabilities jointly shape hierarchical decomposition. The task structure must preserve the contribution of intermediate objectives to the overall goal and the dependencies among them, while the available capabilities determine how far those objectives need to be refined and under what conditions they can be executed together. Decomposition therefore involves coordinating subgoal boundaries, levels of detail, and execution responsibilities. Different choices can yield alternative ways of organizing and accomplishing the same task. Building on these considerations, the next section examines how such alternatives are explored, evaluated, and selected.

4.3 Plan Search and Candidate Selection

A task decomposition can admit multiple ways of achieving its intermediate objectives. Search explores these alternatives, while candidate selection determines which should be retained, refined, or executed. Their interaction governs how an agent allocates computation before committing to a solution. We first review these mechanisms in LLM and coding systems, then examine how physical feasibility and prospective observations shape their embodied use.

Search and selection in LLM and coding systems. Exploring alternatives requires deciding where to introduce variation and when to evaluate it. Complete-solution sampling generates multiple attempts for subsequent selection, whereas intermediate branching compares different continuations before a reasoning trajectory is finished. Sampling and filtering in AlphaCode [231] and thought-level exploration in Tree of Thoughts [407] illustrate these arrangements. The central choice concerns how computation is distributed across alternatives: exploring additional attempts, extending a promising direction, or revisiting an earlier decision. Intermediate evaluation can redirect this effort, but requires estimating the value of a candidate before its eventual

outcome is known. Candidate selection depends on the evidence available for making these judgments. Model-generated assessments can estimate a candidate's promise, while execution results reveal whether particular behavioral requirements are satisfied. CodeT [77] uses generated tests and execution agreement to select code candidates; CodeTree [220] additionally combines execution feedback with model critiques to guide refinement, abandonment, and acceptance. These mechanisms connect evaluation to different decisions within search. Their reliability remains bounded by the coverage and correctness of the tests and by the quality of model judgments. Passing available tests supports a candidate without establishing that it satisfies the full task specification. For interactive agents, evaluation also shapes the actions through which a solution is developed. Gathering information, modifying an artifact, and testing a hypothesis can change which continuations are worth pursuing. LATS [447] incorporates environment feedback and reflection into search across reasoning and interactive tasks, while SWE-Search [40] explores repository-level software-engineering actions using execution information and model-based evaluation. Feedback therefore influences both the preferred candidate and the direction of further exploration. Reconsidering an earlier branch also requires preserving its context and accounting for any changes already made to the environment. These connections between search, evidence, and environment state become particularly consequential when actions affect the physical world.

Search and selection in embodied systems. Embodied search must evaluate whether a candidate advances the task, is currently executable, and leaves suitable conditions for subsequent actions. These considerations connect immediate selection to longer-horizon completion: an individually feasible action may still lead to an unsuccessful continuation. Skill-level selection in SayCan [28] and sequence-level feasibility planning in Text2Motion [236] illustrate these different scopes of evaluation. Accounting for downstream consequences motivates predicting or simulating alternatives before execution. Prospective outcomes allow planners to compare possible continuations and revise choices before committing to physical interaction, as illustrated by VLP [104] and Reflective Planning [113]. This comparison remains provisional because actual outcomes may differ from predictions. New observations must therefore inform subsequent decisions. Receding-horizon planning, exemplified by ParallelWorld [81], connects prospective evaluation with execution feedback, allowing the preferred continuation to change as the environment evolves.

4.4 Plan Representation

Plan representation determines how high-level reasoning is communicated to an execution system and what can be checked before action. We organize this interface into three broad families: *language-based planning*, *Code as Policies (CaP) and programmatic planning*, and *VLA as Tools and skill-based planning*. These families emphasize language-mediated task specification, generated programs, and the orchestration of callable policies, respectively. They can coexist in one system: a program may invoke a VLA, and a language subgoal may parameterize a skill. Symbolic operators, preconditions, and spatial constraints are discussed within these families as ways to structure and validate plans.

Language-based planning. Language-based representations express task intent through semantically meaningful actions and subgoals. Their main advantage is a shared vocabulary for instructions, scene descriptions, and execution feedback, allowing high-level decisions to be revised as observations change. Environment-grounded plans in LLM-Planner [333] and language-mediated feedback in Inner Monologue [**?**] illustrate these complementary uses of language. At this level of abstraction, execution modules must still resolve object references, interpret the requested behavior, and determine whether the intended outcome has been achieved. The central trade-off is between expressive flexibility and explicit execution semantics. Natural language can describe diverse goals without a fully specified action model, while formal compilation makes selected conditions available to a solver. The language–PDDL interface in LLM+P [237] illustrates this transition: task instances are translated under a supplied domain that defines action preconditions and effects. Such compilation connects language to formal planning, with validity depending on both the translated specification and its correspondence to the physical environment.

Code as Policies and programmatic planning. Programmatic representations make the organization of execution explicit. Variables bind task entities to observations, function calls invoke available capabilities, and control flow specifies sequencing, repetition, and conditional responses. Code as Policies (CaP) [234] and ProgPrompt [328] exemplify this use of programs to compose robot behavior. The resulting interface can

Table 4 From task-conditioned policy outputs to physical control. The diagram traces how multimodal observations and a task goal or subgoal condition an action policy, whose representation defines the control space, reference frame, and temporal horizon of its predictions. Discrete, continuous, diffusion-based, and flow-matching generators provide different ways to model this mapping; these generation choices interact with the action representation rather than replacing it. The control layer then converts predicted actions into robot motion, coordinating receding-horizon feedback, policy and motor-control rates, and asynchronous inference to balance continuity with responsiveness. The resulting observations return to the next policy or planning step, so execution is not merely playback of a plan but the point at which its assumptions are tested and corrected through physical interaction.

Family	Variant	Plan interface	Representative work
Language-based	Language plans	Subgoals and action descriptions	LLM-Planner [333]
	Symbolic compilation	Language–PDDL translation	LLM+P [237]
Code-based	Program composition	Control flow, calls, checks	CaP [234]
	Solver-based specifications	Task and geometric constraints	AutoTAMP [82]
Skill-based	Skill-library selection	Skill choice and grounding	SayCan [28]
	Frozen VLA tools	VLA and analytic primitive calls	Harness VLA [438]
	Heterogeneous policies	Capability routing and handoffs	RoboHarness [151]

express how perception and action interact, including where state checks or recovery steps enter execution. Its usefulness depends on the meaning and reliability of the underlying APIs: a well-formed program can still refer to the wrong object or request an infeasible action. A broader computational interface specifies desired outcomes and constraints, delegating action synthesis to a planner or optimizer. This route spans temporal-logic specifications in AutoTAMP [82], spatial value maps in VoxPoser [155], and keypoint constraint functions in ReKep [153]. The common principle is to expose task requirements in a form that supports computational planning. Within this broader family, direct programs prescribe execution structure, whereas specifications define goals, costs, or constraints from which actions are computed. Both require the generated representation to preserve task meaning and remain grounded in the observed scene.

VLA as Tools and skill-based planning. Skill-based representations express a plan through calls to available execution capabilities, each of which realizes behavior over multiple control steps. Low-level action generation is delegated to the selected skill, while planning determines which capability to invoke, with what target or arguments, and in what execution context. The interface can encompass pretrained skill libraries, as in SayCan [28], frozen VLA tools composed with analytic primitives, as in Harness VLA [438], and heterogeneous policy collections, as in RoboHarness [151]. VLA-as-tool planning is thus one instance of the broader problem of composing callable robot capabilities. The key representational issue is how much the planner can know about each capability and its execution outcome. A skill name alone provides limited evidence about when it will succeed or what state it will leave behind. A useful interface should therefore convey applicability, expected effects, and termination or failure feedback. For learned policies, these properties may be estimated from value functions or execution experience rather than specified as fixed symbolic rules. Composition then requires considering both individual skill suitability and whether the outcome of one skill supports the next.

Across these families, the planning value of a representation lies in making execution conditions available to reasoning: grounded goals for language plans, valid calls and constraints for programs or specifications, and capability information for policy tools. These interfaces should also return outcome feedback to support subsequent decisions. Decomposition, search, and representation are coupled parts of embodied planning. Decomposition proposes subgoals; search compares their orderings and consequences; representation grounds the selected branch in robot skills and enables checks; execution supplies the observations that shape the next planning step. Long-horizon performance depends on maintaining these connections as the robot acts and encounters unexpected outcomes.

5 Action Execution

Action execution serves as a critical bridge between high-level decision-making and physical robot interaction in agentic embodied system. While high-level reasoning and planning determine task objectives and intermediate

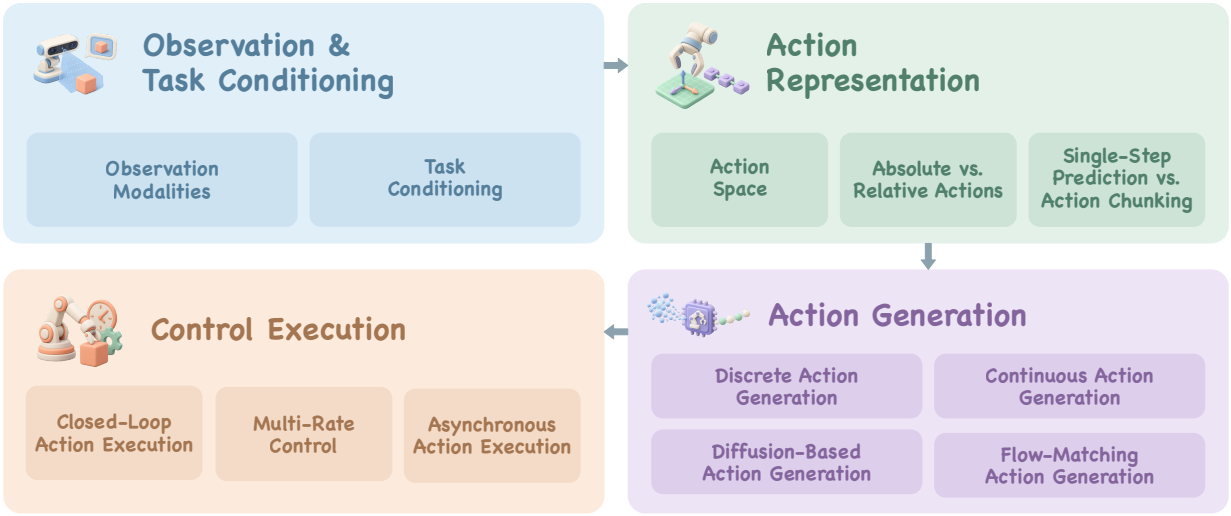


Figure 6 caption

subgoals, action execution translates these decisions into concrete robot control commands. Unlike high-level planning, action execution requires continuously adapting robot actions according to sensory feedback and the current robot state. Recent advances in robot policy learning, ranging from visuomotor policies [67, 86, 279] to large-scale vision-language-action (VLA) models [60, 64, 196], have enabled increasingly generalizable mappings from sensory observations and task conditions to executable robot actions. These policies typically operate over short temporal horizons and can be formulated as:

$$\pi_\theta : (\mathbf{o}_{t-k:t}, \mathbf{g}_t) \rightarrow \mathbf{a}_{t:t+H-1}, \tag{2}$$

where $\mathbf{o}_{t-k:t}$ denotes recent multimodal observations, $\mathbf{g}_t$ the task instruction or subgoal, and $\mathbf{a}_{t:t+H-1}$ a sequence of future actions. This formulation supports diverse action representations and generation paradigms, including autoregressive prediction, continuous regression, diffusion, and flow matching. Beyond action prediction, effective execution requires translating generated actions into low-level control commands to ensure reliable physical interaction. In the following, we review four key aspects of action execution: observation and task conditioning, action representation, action generation, and control execution.

5.1 Observation and Task Conditioning

Short-horizon action policies leverage multimodal observations and task conditions to generate physically feasible actions aligned with task objectives. These inputs can be broadly categorized into two types: *Observations*, which capture the surrounding environment and robot states, and *Task Conditions*, which specify the intended behavior.

Observation Modalities. Visual observations, particularly RGB images, serve as the primary sensory input for modern robot policies, capturing scene appearance, object configurations, and spatial relationships [66, 196]. For geometry-aware manipulation, some approaches incorporate explicit 3D information through depth maps, point clouds, or voxel representations [296, 323, 443]. Proprioceptive observations, including joint positions, velocities, end-effector poses, and gripper states, provide information about the robot current configuration and motion, and are widely incorporated into policies for action prediction [60, 86, 278, 279, 442]. Force and tactile observations provide direct feedback on physical contacts and interaction dynamics that are difficult to infer from vision alone, enabling more precise and adaptive action generation in contact-rich manipulation [58, 179, 413, 440].

Task Conditioning. Task conditions specify the desired behavior of robot policies and can take various forms, including natural-language instructions, goal images, and intermediate subgoals. *Language conditioning* has

Table 5 Observation modalities and task conditioning in representative robot manipulation policies.

Category	Input	PerAct [323]	OpenVLA [196]	Octo [279]	ForceVLA [413]	FuSe [179]	Goal-VLA [80]	CoT-VLA [441]	$\pi_{0.5}$ [59]
Observation Modalities	RGB	✓	✓	✓	✓	✓	✓	✓	✓
	3D Geometry	✓	–	–	–	–	✓	–	–
	Proprioception	✓	–	✓	✓	✓	✓	–	✓
	Force/Tactile	–	–	–	✓	✓	–	–	–
	Audio	–	–	–	–	✓	–	–	–
Task Conditioning	Language	✓	✓	✓	✓	✓	✓	✓	✓
	Goal Image	–	–	✓	–	–	✓	✓	–
	Subgoal	–	–	–	–	–	–	✓	✓

become widely adopted in VLA models, enabling policies to interpret semantic instructions and generalize across diverse manipulation tasks [60, 65, 66, 196, 278]. *Goal-image conditioning* provides visual representations of desired task outcomes, either as direct policy inputs [279] or as generated goal states that guide downstream manipulation [80]. **Subgoal conditioning** is commonly employed in hierarchical frameworks, where high-level planners decompose complex tasks into intermediate objectives or executable skills that guide low-level action policies [29, 154, 423]. Table 5 summarizes the observation modalities and task conditions adopted by representative robot policies.

After selecting appropriate observations and task conditions, action policies integrate these heterogeneous inputs into a unified representation for action generation. Transformer-based architectures have become a widely adopted solution, leveraging modality-specific embeddings and attention mechanisms to capture cross-modal dependencies [60, 278, 279]. Through such interactions, task information can guide the extraction of relevant sensory features, while robot proprioceptive state provides the physical context for action prediction. In this way, multimodal fusion bridges sensory perception and task objectives, providing a unified conditioning representation for subsequent action generation.

5.2 Action Representation

Action representation defines how robot motions are parameterized and organized for policy prediction. Existing short-horizon action policies differ in their control spaces, reference conventions, and temporal horizons, each affecting the flexibility, generalization, and execution efficiency of learned policies.

Action Space. Robot actions are commonly represented in joint space or task space, depending on the control interface. *Joint-space actions* specify joint positions, velocities, or torques, enabling explicit control of robot configurations and coordinated multi-joint motions, as demonstrated by ACT [442] and Mobile ALOHA [118]. While closely aligned with robot actuation, these representations are often embodiment-specific due to differences in joint structures and degrees of freedom. In contrast, *task-space actions* describe end-effector poses, Cartesian displacements, or velocities, typically accompanied by gripper commands, as adopted by RT-1, RT-2, RT-X, and OpenVLA [65, 66, 196, 282]. Such representations provide a more transferable geometric interface across robot embodiments, but generally require inverse kinematics or low-level controllers to satisfy joint limits, reachability, and collision constraints. To accommodate heterogeneous robots and control requirements, recent generalist policies, such as Octo [279] and RDT-1B [243], further support diverse action spaces, including joint-space and end-effector-based representations.

Absolute vs. Relative Actions. Beyond the control space, robot actions can be represented as absolute targets or relative changes. *Absolute actions* specify desired joint configurations or end-effector poses in a predefined reference frame, providing explicit control targets without accumulating errors from successive action integration [86, 442]. However, predicting absolute targets may require learning complex mappings from observations to global robot configurations. In contrast, *relative actions* predict incremental joint displacements or Cartesian motions with respect to a reference state, facilitating local motion adjustments and closed-loop control [66, 196, 279]. While relative representations can simplify action prediction and adapt to varying initial configurations, accumulated prediction or tracking errors may cause drift during extended execution. Recent empirical studies further highlight the trade-offs between absolute and relative

representations in terms of policy learnability and control stability [112]. Ultimately, the choice depends on task precision, execution horizon, and the underlying control interface.

Single-Step Prediction vs. Action Chunking. Beyond spatial parameterization, robot actions also differ in their temporal horizons. *Single-step policies*, such as RT-1, RT-2, and OpenVLA [65, 66, 196], predict an individual action $\mathbf{a}_t$ at each timestep, enabling frequent incorporation of new observations and rapid adaptation to environmental changes. However, repeated policy inference introduces computational overhead, while independently predicted actions may lead to temporally inconsistent motions. In contrast, *action-chunking policies* jointly predict a sequence of future actions $\mathbf{a}_{t:t+H}$, as adopted by ACT [442], Diffusion Policy [86], Octo [279], π_0 [60], and GR00T N1 [278]. By modeling temporal dependencies across consecutive actions, chunking facilitates smoother and more coordinated motions while reducing inference frequency when multiple actions are executed per prediction. Nevertheless, executing longer action chunks without intermediate feedback can limit responsiveness and increase sensitivity to unexpected disturbances. In practice, the prediction and execution horizons can be decoupled through receding-horizon control, allowing policies to balance temporal consistency, computational efficiency, and closed-loop adaptability.

5.3 Action Generation

Given multimodal observations and task conditions, action generation models learn to predict executable robot actions according to the chosen action representation. Existing approaches differ primarily in how they model the conditional action distribution, ranging from discrete token prediction and direct continuous regression to generative diffusion and flow-matching models. These approaches exhibit different trade-offs in action expressiveness, computational efficiency, and the ability to model multimodal action distributions. Table 6 summarizes the representative methods, generation mechanisms, and key advantages and limitations of these paradigms.

Discrete Action Generation. Discrete action generation converts continuous robot actions into discrete tokens, enabling action prediction through transformer-based sequence modeling. Early approaches, such as Gato [303] and RT-1 [66], demonstrate this paradigm by discretizing action dimensions into predefined bins. Building on pretrained vision-language models, RT-2 [65] and OpenVLA [196] further formulate action generation as autoregressive token prediction, unifying language understanding and robot control within a shared framework. However, conventional discretization can introduce quantization errors, while autoregressive decoding incurs increasing computational costs for high-dimensional actions and long action sequences. To overcome these limitations, SpatialVLA [296] introduces adaptive spatial action grids to improve the representation of 3D motions, whereas FAST [287] employs frequency-based tokenization to compress continuous action sequences into compact discrete representations, reducing autoregressive decoding overhead.

Continuous Action Generation. Direct continuous-action policies predict real-valued robot actions without discrete tokenization or iterative sampling. Deterministic regression approaches typically optimize supervised objectives such as ℓ_1 or ℓ_2 losses. Early methods, such as BC-Z [164], learn continuous control commands through behavior cloning, while GR-1 [378] and RoboFlamingo [228] incorporate pretrained visual or vision-language representations for continuous action prediction. OpenVLA-OFT [194] further enables parallel action regression, improving inference efficiency compared with autoregressive token decoding. However, when multiple valid actions exist for the same observation, deterministic regression may average different behaviors, potentially producing ineffective motions. To address this limitation, latent-variable approaches, such as Play-LMP [251], HULC [267], and ACT [442], introduce learned latent representations to capture variations in demonstrated behaviors. Overall, these methods enable efficient, quantization-free action prediction without iterative sampling, but their ability to capture diverse valid behaviors remains limited, particularly for deterministic regression-based action policies.

Diffusion-Based Action Generation. Diffusion-based policies model conditional action distributions by progressively denoising randomly initialized action sequences into continuous robot trajectories. Given an action sequence $\mathbf{a}$ and conditioning information $\mathbf{c}$, the model is typically trained to predict the noise added at different diffusion steps, using the following objective:

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{\mathbf{a},\boldsymbol{\epsilon},s}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_s, s, \mathbf{c})\|_2^2\right], \tag{3}$$

Table 6 Comparison of action generation paradigms in representative robot policies.

Category	Representative Methods	Generation Mechanism	Advantages	Limitations
Discrete Action Generation	RT-1 [66], RT-2 [65], OpenVLA [196], SpatialVLA [296], FAST [287]	Discrete action tokenization and sequence prediction	Unified VLM-action modeling; leverages pretrained language models	Quantization errors; sequential decoding overhead
Continuous Action Generation	BC-Z [164], GR-1 [378], RoboFlamingo [228], ACT [442], OpenVLA-OFT [194]	Direct continuous regression or latent-variable prediction	Quantization-free actions; efficient and parallel prediction	Mode averaging in deterministic regression; latent modeling complexity
Diffusion-Based Action Generation	Diffusion Policy [86], DP3 [424], MDT [307], Octo [279], RDT-1B [243]	Iterative denoising of continuous action trajectories	Multimodal action distributions; flexible trajectory generation	Multiple denoising steps; inference latency
Flow-Matching Action Generation	π_0 [60], $\pi_{0.5}$ [59], GR00T N1 [278], SmolVLA [325], X-VLA [444]	Continuous velocity-field learning and numerical integration	Multimodal action distributions; potentially efficient ODE sampling	Numerical integration overhead; quality–step trade-off

where $\mathbf{x}_s = \sqrt{\bar{\alpha}_s}\mathbf{a} + \sqrt{1-\bar{\alpha}_s}\boldsymbol{\epsilon}$ denotes the noisy action at diffusion step s, and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Unlike deterministic regression, diffusion models can capture multiple valid action modes and jointly generate temporally coordinated action sequences, making them suitable for complex manipulation tasks [86]. Building on Diffusion Policy [86], DP3 [424] incorporates 3D geometric representations to improve visuomotor generalization, while MDT [307] extends diffusion transformers to multimodal goal conditioning. At a larger scale, Octo [279] enables generalist action generation across heterogeneous robot datasets, and RDT-1B [243] scales diffusion transformers for bimanual manipulation with unified action representations. Despite their expressiveness, diffusion policies typically require multiple denoising steps, introducing computational overhead and a trade-off between sampling quality and inference latency.

Flow-Matching Action Generation. Closely related to diffusion models, flow matching provides an alternative generative framework that learns a time-dependent velocity field to transform random noise into continuous action trajectories. Given an action sequence $\mathbf{a}$ and conditioning information $\mathbf{c}$, a common conditional flow-matching objective is defined as:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{\mathbf{a},\boldsymbol{\epsilon},s}\left[\left\|\mathbf{v}_\theta(\mathbf{x}_s, s, \mathbf{c}) - (\mathbf{a} - \boldsymbol{\epsilon})\right\|_2^2\right], \tag{4}$$

where $\mathbf{x}_s = (1-s)\boldsymbol{\epsilon} + s\mathbf{a}$ linearly interpolates between Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and target actions $\mathbf{a}$, with $s \in [0, 1]$. During inference, action trajectories are generated by numerically integrating the learned velocity field from noise to the target action distribution. This paradigm has been adopted by generalist VLA models, including π_0 [60] and $\pi_{0.5}$ [59], which integrate flow-matching action experts with pretrained vision-language representations. GR00T N1 [278] extends this framework to humanoid manipulation, while SmolVLA [325] and X-VLA [444] emphasize efficient deployment and cross-embodiment generalization, respectively. Like diffusion models, flow matching can capture complex, multimodal action distributions, while potentially enabling efficient generation with fewer integration steps. However, inference still requires numerical integration, introducing a trade-off between action quality and computational efficiency that depends on the integration scheme and model architecture.

5.4 Control Execution

Beyond action generation, reliable robot manipulation requires translating predicted actions into physical execution while maintaining responsiveness to environmental changes. Closed-loop execution continuously updates actions based on new observations, while multi-rate control coordinates the different operating frequencies of learned policies and robot controllers. Asynchronous execution further reduces inference-induced delays by overlapping action prediction with ongoing robot motion, whereas low-level motor controllers convert predicted actions into executable robot commands. Together, these mechanisms bridge learned action policies and real-world robot execution, enabling responsive and stable manipulation.

Closed-Loop Action Execution. Closed-loop execution enables robot policies to continuously adapt their actions based on new observations, correcting deviations caused by prediction errors or environmental disturbances.

A common strategy is **receding-horizon control**, which executes only part of a predicted action chunk before replanning with updated observations [86, 279]. This strategy balances temporal consistency with responsiveness to environmental changes. However, frequent replanning may produce inconsistent predictions across successive action chunks. To address this issue, ACT [442] introduces **temporal ensembling**, which averages overlapping action predictions from successive chunks to improve motion smoothness.

Multi-Rate Control. Real-world robot systems operate at multiple temporal frequencies, with learned policies typically running at lower rates than low-level motor controllers [60, 66, 86]. This mismatch is particularly pronounced in large VLA models, where inference latency may exceed the robot control period [194, 196]. To bridge this gap, action-chunking policies, such as π_0 [60] and OpenVLA-OFT [194], generate multiple future actions in a single inference step, allowing commands to be buffered and executed at higher frequencies. When necessary, interpolation or resampling further aligns predicted actions with the low-level control frequency. By decoupling policy inference from high-frequency motor control, multi-rate execution enables computationally expensive policies to provide continuous control references without requiring inference at every control step.

Asynchronous Action Execution. Despite action chunking, synchronous execution may introduce pauses when the robot must wait for the next action prediction, particularly for computationally expensive VLA models. To address this limitation, **asynchronous execution** overlaps policy inference with ongoing robot motion, allowing subsequent action chunks to be generated before the current ones are exhausted. SmolVLA [325] implements this paradigm through asynchronous inference and action buffering to maintain continuous robot execution. However, independently generated action chunks may introduce abrupt motion transitions, while inference delays can cause discrepancies between the observed and actual robot states. To improve execution continuity, Real-Time Chunking (RTC) [61] conditions new action predictions on previously committed commands through action inpainting. FutureRTC [172] further addresses temporal misalignment by predicting execution-time observations and robot states. Together, these approaches improve the consistency of asynchronous execution across successive action chunks and changing robot states.

6 Closed-Loop Evolution

The preceding chapters decompose embodied intelligence into a perception, planning, and action loop. Yet, treating these components as static, feedforward primitives leaves the robot inherently brittle, where the system stagnates upon encountering out-of-distribution physical perturbations in open world deployment. While this continuous self-improvement cannot be realized within classical open-loop pipelines or monolithic end-to-end policies, it fundamentally demands the AEI framework. A vital conceptual blueprint for this physical agency emerges from the digital domain. Autonomous coding agents have established a mature paradigm of closed-loop iteration: synthesizing code, executing in isolated sandboxes, inspecting runtime tracebacks, iteratively debugging, and committing validated functions to reusable libraries. Within the AEI framework, this software engineering loop is mapped onto physical reality: sensory anomalies function as physical tracebacks, motor failures trigger in-situ programmatic debugging, and newly acquired manipulation primitives are registered as persistent tools.

As illustrated in Fig. 7, coding and embodied agents share a feedback-driven evolution loop, but differ in reset cost, feedback verifiability, and the gap between validation and deployment. Coding environments typically support inexpensive resets, explicit test outcomes, and reproducible execution, whereas physical interaction introduces costly scene restoration, ambiguous sensor evidence, and a sim-to-real gap. These differences make it harder to establish whether a successful recovery will generalize beyond the current trial. The comparison therefore motivates a distinction between correcting the current execution and validating a persistent update for future tasks.

This chapter develops that view in four steps. Section 6.1 distinguishes the two time scales of change and explains five possible update targets, together with the evidence needed before an update can be called an improvement. Section 6.2 describes how coding agents currently execute, verify, revise, and reuse changes. Section 6.3 reviews the corresponding mechanisms in embodied systems, from within-episode recovery to policy and data-pipeline adaptation. Section 6.4 then identifies the components that are still missing, proposes a practical pipeline for connecting them, and derives the main evaluation and research challenges.

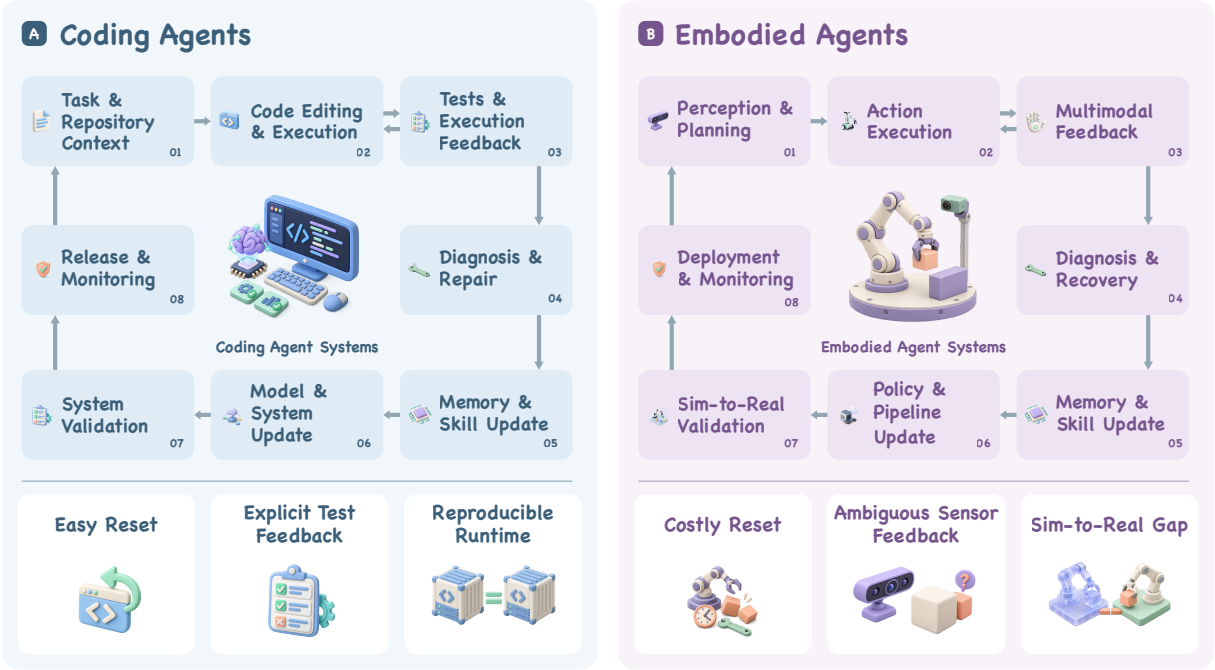


Figure 7 Conceptual comparison of evolution loops in coding and embodied agent systems. (A) Coding agents use execution and test feedback to diagnose failures, refine memory and skills, and inform model and system updates before validation and release. (B) Embodied agents use multimodal feedback to guide recovery and updates to memory, skills, policies, and pipelines, with sim-to-real validation and deployment monitoring. The bottom panels highlight three practical contrasts: reset cost, feedback verifiability, and the gap between validation and deployment. These contrasts describe typical operating conditions; the update stages may involve external orchestration or human oversight.

6.1 Evolution Targets

6.1.1 How Execution Feedback Becomes a Persistent Update

This chapter uses five names for the place where a change is made: *control*, *reflection*, *memory*, *models*, and *system*. They describe persistence and evidence requirements rather than a universal maturity ladder. One agent can use several targets in the same episode, and a small, well-tested memory update can be more reliable than a large but unverified model change.

The *control* target changes the next action or a short local trajectory. A grasp that is corrected after a measured slip is a control correction; the evidence is recovery within the current episode. The *reflection* target changes the agent's explanation of the failure or its current plan. It should be counted only when the revised explanation leads to a better intervention under the same retry budget. The *memory* target stores an experience, fact, or executable skill for later use, so the claim requires provenance and improvement on new instances after transient context has been removed. The *models* target changes a policy, value function, or world model using new data; it requires held-out gains and checks that earlier abilities are retained. The *system* target changes the agent's code, tools, curriculum, reward, or update procedure. This is the broadest claim and needs independent comparisons across versions.

Control and reflection normally operate within one episode, while memory, models, and system changes persist across episodes. The boundary is functional rather than architectural: a language model can support control, and a conventional learner can provide model adaptation. The agentic contribution is the selection of informative evidence, the choice of an appropriate update target, and the decision to accept or reject the resulting change. Changing a tool or update procedure alone does not establish recursive self-improvement unless it also improves the system's later ability to generate and validate improvements.

6.2 How Coding Agents Implement Closed-Loop Improvement

Coding agents offer a useful reference because repository actions have executable consequences and inexpensive verifiers. The environment is a versioned workspace; observations include tests, compiler traces, logs, and diffs. This makes the loop cheap and reproducible, but does not by itself establish unrestricted self-improvement.

6.2.1 Control and Reflection: Execute, Verify, and Replan

At the control and reflection targets, a coding agent makes a bounded edit, runs a relevant check, interprets the result, and chooses the next edit. SWE-bench supplies issue-level tasks and test-based assessment [177]; SWE-agent and OpenHands show how the agent–computer interface and isolated workspaces expose actions and feedback [366, 398]. ReAct, Self-Refine, Reflexion, and CRITIC instantiate action–observation interleaving, critique, textual experience, and tool-assisted checking [125, 255, 321, 408]. Reflection counts as correction only when it changes a checkable patch or test. Self-consistency and intrinsic self-correction are inference-time baselines rather than persistent updates, and can degrade without reliable external feedback [150, 367]. SWE-Search adds search over repair trajectories; its gains should be called persistent only if they transfer to later tasks [39].

6.2.2 Memory: Reuse a Validated Procedure Across Tasks

At the memory target, a successful repair, command sequence, or failure pattern becomes a retrievable asset. Voyager illustrates this mechanism through an executable skill library and automatic curriculum in Minecraft [355]. A memory update is supported only when transient context is cleared and the asset helps on new tasks under an applicability check. Repository changes can still invalidate a command or cause overfitting to visible tests, so provenance and expiry matter even in software environments.

6.2.3 Models and Systems: Adapt the Agent or Its Training Process

At the models target, interaction data updates a policy or value model; at the system target, the agent changes its code, tools, search procedure, reward, or curriculum. The Darwin Godel Machine exemplifies bounded system modification: it searches over agent-code variants, evaluates them on coding benchmarks, and retains candidates in an archive [432]. Its results support system changes under the reported safeguards and budget, rather than unrestricted recursive self-improvement.

Coding-agent loops are comparatively mature because snapshots provide cheap resets, diagnostics are informative, specifications are executable, and versions are easy to compare. These are requirements for a credible embodied loop, not evidence that coding agents have solved self-improvement.

Table 7 shows the common pattern: an execution signal is interpreted, a targeted change is proposed, and later trials decide whether it should be reused. Coding agents can usually reset to a known snapshot, inspect a precise trace, and compare versions cheaply. Embodied agents have counterparts for all five targets, but occlusion, contact uncertainty, irreversible actions, embodiment differences, and hardware drift make the same evidence much harder to obtain.

6.3 Embodied Closed-Loop Evolution: Current Approaches

Embodied systems contain counterparts to all five update targets described above, but the evidence and failure modes change when actions affect a physical, partially observed world. The following subsections review current approaches in increasing order of persistence, from within-episode recovery to policy and training-pipeline updates.

6.3.1 Within-Episode Control Correction: Verify Actions and Recover

At the within-episode control-correction target, the system changes the next action after observing execution rather than changing its persistent policy. Feedback control and reinforcement learning provide the foundation for adapting actions to measured outcomes [199]. SayCan grounds language-based skill selection in learned affordance estimates [27]; Inner Monologue feeds success detection, scene descriptions, and human interaction

Table 7 Representative closed-loop mechanisms in digital and embodied agents. Each row names the main update target and the capability shared by the two settings.

Target	Digital-agent examples	Embodied-agent examples	Shared capability
Control	SWE-agent [398]; OpenHands [366].	SayCan [27]; Inner Monologue [156]; Code as Policies [234].	Feedback-guided action correction.
Reflection	ReAct [408]; Self-Refine [255]; Reflexion [321]; CRITIC [125].	REFLECT [247].	Failure interpretation and replanning.
Memory	Reflexion [321]; Voyager [355].	Voyager [355]; PhySGA [252]	Reuse of prior experience.
Models	SWE-Search [39]; Darwin Godel Machine [432].	RoboCat [63]; RT-1 [67]; RT-2 [64]; Open X-Embodiment [280].	Data-driven model adaptation.
System	Darwin Godel Machine [432].	Eureka [254]; GenSim [360]; RoboGen [370].	Search over update procedures.

back into language-based planning [156]; Code as Policies exposes generated behavior through robot APIs [234]. These systems improve the connection between semantic intent and executable action, but a successful recovery does not by itself create a new skill.

Verification must combine task predicates with sensor evidence at an appropriate temporal resolution. For a pick-and-place operation, relevant evidence may include object identity before contact, gripper closure and load during lifting, object motion relative to the gripper, and the final placement relation. A final image alone may fail to distinguish a stable placement from an object that is about to fall. Conversely, low-level execution success may conceal a semantic error such as moving the wrong item. The monitor should retain the provenance and confidence of each predicate rather than collapse all feedback into a single textual success judgment.

6.3.2 Task-Level Reflection: Diagnose Failure and Replan

At the task-level reflection target, an embodied agent converts a failed or ambiguous outcome into a causal hypothesis and a revised plan. REFLECT constructs hierarchical summaries of multisensory robot experience, asks a language model to explain the failure, and passes the explanation to a planner for correction; its RoboFail dataset evaluates failure scenarios and corrective planning [247]. This is a direct physical counterpart to tool-supported reflection, but the observations must first be grounded in contact, motion, and scene events.

A diagnostic record should separate symptom, hypothesis, evidence, and proposed test. "The cup did not move" is a symptom; "the grasp slipped" is a hypothesis; a tactile change or camera observation is evidence; and a different grasp followed by load verification is an intervention. Without these distinctions, an untested explanation can become a persistent rule. Hierarchical summaries reduce context length but may discard the transient contact event needed to distinguish two causes, so summaries should link back to timestamped raw observations. A language explanation remains a hypothesis unless recovery, intervention, or controlled perturbation supplies evidence for it.

6.3.3 Persistent Memory and Skills: Reuse Experience Across Episodes

At the persistent-memory target, an embodied system turns a recovery or successful trajectory into a reusable fact, program, or skill. Voyager demonstrates executable skill accumulation in a virtual embodied environment [355]; Reflexion demonstrates reuse of textual experience across attempts [321]; PhySGA distills simulator trial-and-error into a reusable skill library [252]. Physical transfer requires a richer applicability condition because a skill can depend on end-effector geometry, payload, material, calibration, and available sensors.

Table 8 Capability coverage in representative software and embodied closed-loop systems. A checkmark means routine support; a triangle means partial or task-dependent support; a cross means that the capability is generally not established.

Capability	Software	Embodied	Interpretive implication
Observable execution trace	✓	△	Sensors are incomplete under occlusion and contact.
Cheap reset and reproducible state	✓	×	Physical trials consume time, hardware, and sometimes objects.
Cheap rollback and state restoration	✓	△	A physical failure can damage the scene or the robot.
Failure diagnosis with external checks	✓	△	Physical symptoms often have several plausible causes.
Persistent memory or skill reuse	✓	△	Transfer needs explicit embodiment and context conditions.
Independent held-out validation	✓	△	Simulation and learned evaluators may share the candidate's blind spots.
Version tracking and independent promotion	✓	△	Calibration and hardware drift complicate safe release.

A useful embodied memory entry can be represented as

$$m = (c, \xi, \kappa, \rho, v), \tag{5}$$

where c specifies the applicability context, ξ links to supporting experience, κ stores a fact or executable skill, ρ records confidence and validation status, and v identifies the source system and embodiment. This schema is a design synthesis, not a claim that the cited systems all implement it. Retrieval should test c against the current state; consolidation should resolve conflicting entries and retire stale ones. Memory quality is measured by downstream decisions: whether retrieval helps on new instances, whether irrelevant memories are rejected, and whether removing the memory reverses the claimed gain.

6.3.4 Policy and World-Model Adaptation: Learn from New Interaction Data

At the policy-adaptation target, the system uses demonstrations or self-generated trajectories to train a new policy. RoboCat combines heterogeneous robotic experience with task and embodiment adaptation and shows how the trained agent can generate data for later training iterations [63]. This supports an empirical data-improvement loop on the tested tasks and embodiments; it does not establish autonomous failure diagnosis or universally beneficial self-training.

RT-1 and RT-2 provide scalable mappings from observations and instructions to actions [64, 67]; PaLM-E integrates continuous embodied inputs into a language-model-based reasoning system [103]. Open X-Embodiment supports cross-embodiment data and model transfer, Octo provides an open generalist policy, and DROID supplies diverse real-world manipulation trajectories [191, 279, 280]. These are enabling models and resources rather than, by themselves, autonomous deployment–update–validation cycles. Repeated collection from the current policy can also reinforce coverage gaps, making targeted data selection and held-out evaluation essential.

6.3.5 System and Pipeline Evolution: Generate Rewards, Tasks, and Training Data

At the system-and-pipeline target, the update is a reward, curriculum, simulator task, or training procedure rather than a single action or policy checkpoint. Eureka generates reward code and uses reinforcement-learning outcomes as feedback for further reward design [254]. GenSim generates simulation tasks and demonstrations, including goal-directed curricula and exploratory task generation [360]. RoboGen organizes a propose–generate–learn cycle that creates tasks, scenes, and training supervision and selects learning methods for subskills [370].

Simulation lowers reset and trial costs, but it also changes what the evidence establishes. High return can exploit an imperfect reward or simulator, and generated tasks can be redundant or unrelated to deployment failures. GenSim includes real-world transfer experiments, but these remain evidence for the tested transfer conditions. Generated rewards and curricula therefore need independent checks for task validity, diversity, reward exploitation, and physical transfer. A simulator is a source of candidate evidence with stated assumptions, not a universal verifier.

Human involvement remains part of this target. Mobile ALOHA demonstrates the value of whole-body teleoperation for learning mobile manipulation [118]; studies of offline human demonstrations show that data quality and learning choices substantially affect manipulation performance [264]; AutoRT uses foundation models to orchestrate robot tasks and data collection [30]. These systems motivate selective data acquisition: collect experience that addresses an observed weakness, record demonstrations and interventions as costs, and test whether the update improves behavior beyond the collection setting.

6.4 From Component Loops to a Complete Embodied Evolution Pipeline

The preceding review identifies useful components, but most embodied results validate only one part of the outer loop at a time. This section turns those components into a concrete argument: first, what is already supported by evidence; second, what a complete pipeline must record and verify; and third, which open problems prevent that pipeline from operating reliably. The comparison with coding agents is used to expose requirements for evidence and release, not to rank the intelligence of the two domains.

6.4.1 Current Evidence and the Remaining Gap

Embodied systems already provide several foundations for evolution. Proprioception, force sensing, geometric constraints, and local controllers expose physical execution. Language-conditioned planners can insert recovery actions; multimodal failure analysis can connect an outcome to scene and contact evidence; executable skills and cross-embodiment datasets provide mechanisms for transfer; and simulation or teleoperation can lower data-collection costs. Inner Monologue and REFLECT show feedback-conditioned replanning, Voyager shows persistent executable-skill reuse, RoboCat shows a policy/data loop, and Eureka, GenSim, and RoboGen show that rewards, tasks, and supervision can be generated or revised [63, 156, 247, 254, 355, 360, 370].

These results remain modular rather than a demonstration of one self-evolving robot. Compared with coding agents, embodied systems still face five linked gaps: state and outcome observability under occlusion and contact; action reversibility when the world can be damaged; failure attribution across perception, planning, calibration, and control; applicability of skills across embodiments and changing objects; and independent promotion of a physical update. A robot therefore needs more than a reflection module: it needs a route from multimodal evidence to a narrow change and an independent release decision.

6.4.2 From a Robot Failure to a Safe System Update

This section explains what must happen between observing a failure and deploying a change. Its purpose is practical: it connects the components reviewed above into a sequence that records evidence, selects a narrow update, tests it, and provides a way back if the update is unsafe. It is a reference workflow for organizing experiments, rather than a claim that current systems already implement every step.

A complete embodied loop can be described as six connected functions:

1. **Record the interaction.** Store synchronized observations, commands, measured execution, subgoal checks, interventions, and software, model, and hardware versions. Summaries should link back to timestamped raw evidence.
2. **Check what happened.** Combine terminal task checks with process signals and safety monitors. Keep uncertainty explicit and distinguish failure from an invalid or inconclusive evaluation.
3. **Diagnose before changing anything.** Separate candidate causes in perception, state estimation, planning, skill selection, control, and interfaces. If the evidence is insufficient, request another observation or a controlled diagnostic action.

4. **Choose a narrow update.** Select the smallest suitable target: invalidate a memory, edit a skill, collect a demonstration, revise a reward or curriculum, or update a policy or world model. Record the supporting evidence and the conditions under which the change is expected to work.
5. **Validate in stages.** Use replay for historical evidence, simulation for counterfactuals under stated assumptions, and constrained physical trials for transfer. Shadow deployment can compare decisions without actuation, but it cannot replace a physical test of the action's outcome.
6. **Promote and monitor.** Register provenance, calibration, models, tools, and evaluation results. Release a candidate only after regression and safety checks, retain a feasible recovery configuration, and feed new failures back into the log.

Candidate generation must be separated from promotion. A model that proposes a change and declares it successful can share the same blind spot, so independent sensors, held-out tasks, counterfactual checks, or human audits are needed. Logged trajectories support replay but generally cannot reveal what a different policy would have done, because changed actions alter future observations. Off-policy estimates therefore require coverage assumptions, and simulation is useful only within the limits of its dynamics and sensing models.

The update should match the diagnosis. A wrong stored object location calls for memory invalidation; repeated grasp failures may justify a skill update; broad failures across objects may require new data and policy adaptation. Promotion should require a measurable benefit, no unacceptable regression or safety violation, and a documented interaction and human-supervision budget. The candidate should remain versioned and reversible so that later failures can be attributed to the change and the previous configuration can be restored.

7 Computing Infrastructure

The capability ceiling of agentic embodied intelligence is no longer determined by models alone. VLA models attain strong single-task performance, yet their deployment encounters two structural obstacles: a *capacity mismatch* between billion-parameter models and palm-sized onboard compute, and an *architectural limitation* of single-shot mappings over long-horizon tasks. The agentic paradigm resolves the second obstacle by decomposing the system into a deliberating model, fast-executing policies, memory, and standardized tool interfaces—and in doing so it multiplies the demands placed on the layer beneath. This chapter therefore treats **computing infrastructure** as the bridge between models and embodiments. As shown in Figure 8, a model is not embodied until its outputs are realized on actuators, and a body remains unproductive until it receives structured intent.

We organize the discussion in four parts. Section 7.1 analyzes how the agentic shift changes the infrastructure requirement, and introduces the four-layer stack used throughout. Section 7.2 studies deployment: how computation is mapped onto device, edge, and cloud, and where the split is placed. Section 7.3 examines the agent runtime and the Model Context Protocol (MCP), which integrate upward into the algorithm stack and downward into the hardware stack, respectively. Section 7.4 briefly reviews hardware and embodiments, and Section 11.5 closes with a gap map of what has been built and what remains open.

7.1 From End-to-End to Agentic: The Infrastructure Requirement

Two quantitative observations anchor the chapter. First, open VLAs now span roughly 1B to 55B parameters [65, 196], whereas robot-onboard compute is an edge module bound by hard power, thermal, and cost budgets. Second, as formalized in Table 1, end-to-end policies operate largely within L2. The transition to L3–L5 requires task decomposition, tool invocation, long-horizon memory, and closed-loop recovery. None of these capabilities resides in the policy network itself: all are carried by the infrastructure layer. *The higher the autonomy level, the more the differentiator is the substrate rather than the model.*

Under the end-to-end paradigm, this layer faced a single question: how to execute one model efficiently. However, under the agentic paradigm, it faces a chain of systems questions concerning computation placement, communication, scheduling, memory, and recovery. Whether inference should reside onboard or be consumed as a hosted service remains itself an open debate [10]. To structure these questions we adopt a four-layer

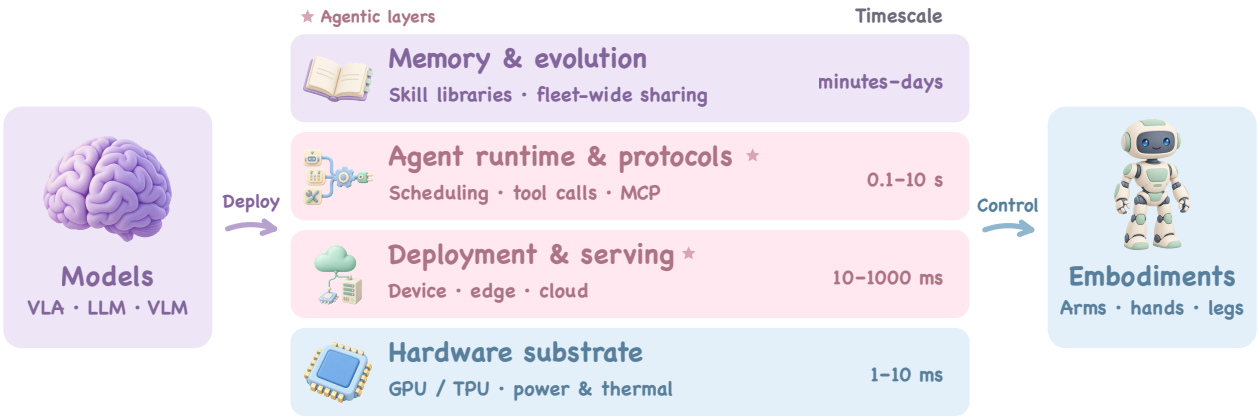


Figure 8 Positioning of infrastructure as the bridge between models and embodiments. It receives algorithms from above and manages hardware from below.

stack with explicit per-layer timescales: hardware, deployment and serving, agent runtime and protocols, and memory. Real-time constraints tighten toward the bottom: control loops operate at millisecond rates, policies at tens of milliseconds, and deliberation at second rates. The two middle layers are the new territory of this chapter; adjacent surveys cover agentic architectures in general [8], LLM serving systems [20], and connected robotics [9, 12], but none systematizes the agentic-embodied infrastructure stack as a whole.

7.2 Deployment across Device, Edge, and Cloud

We first fix terminology: the *device side* is the compute module carried by the robot (e.g., a Jetson-class module); the *edge side* is an on-site server, canonically a single workstation-class machine in a factory; the *cloud side* is data-center-scale accelerators.

The Parameter Scale Gap. The deployment problem originates in a simple comparison: OpenVLA [196] carries approximately 7B parametersx, whereas the lightweight action predictor that executes in real time onboard in VLA-ULAP [21] carries 7.4M, a gap of nearly three orders of magnitude (≈950×). Engineering narrows this gap but does not close it. Onboard compute hosts reflex-scale competence, not full model capability.

Deployment Routes: Onboard, Cloud, and Hybrid. As shown in Figure 9, three deployment. Route A (fully onboard) compresses the model onto the robot. DeeR-VLA [3] selects network depth per task difficulty. HiRT [4] pairs a low-frequency VLM with a high-frequency lightweight policy. FASA [18] adapts the sampling budget of diffusion-based VLAs. Compute, energy, and thermals are paid by the platform, and performance is capped by onboard hardware. Route B (fully cloud) relocates inference wholesale to the data center, the canonical instance being RT-2 [65] with cloud-TPU execution. Cloud robotics itself predates the VLA era. Kuffner's 2010 vision [202], fog robotics [1], and FogROS [2] prefigure today's fleet-level serving. The route is fragile to network jitter, bandwidth, and privacy constraints. Route C (hybrid) combines cloud deliberation with onboard reflexes and is the current research focus [13, 16, 17, 21].

Latency Constraints Across Control Loops. The case for hybrid organization must be made per control loop. The fast loop operates at tens of hertz with a budget of tens of milliseconds: the onboard 7.4M-parameter predictor completes a step in 19.9 ms on a Jetson Orin Nano [21], whereas a cloud round trip already averages ∼100 ms. Crucially, tail latency and disconnection—not mean latency—preclude a purely cloud-fed fast loop. The slow deliberation loop tolerates ∼100 ms comfortably, a property exploited explicitly by HiRT's low-frequency design [4]. VLA-ULAP [21] exemplifies the resulting organization: the cloud VLA is invoked on demand while the onboard predictor emits action chunks—short sequences of future actions produced in a single forward pass—and the two sources interleave. The closed-weight ecosystem answers the same question with a different contract: Gemini Robotics retains closed weights, but users submit task data to a managed supervised-fine-tuning API and receive a specialized policy for their own devices [11]—model recipes are produced centrally while execution remains local.

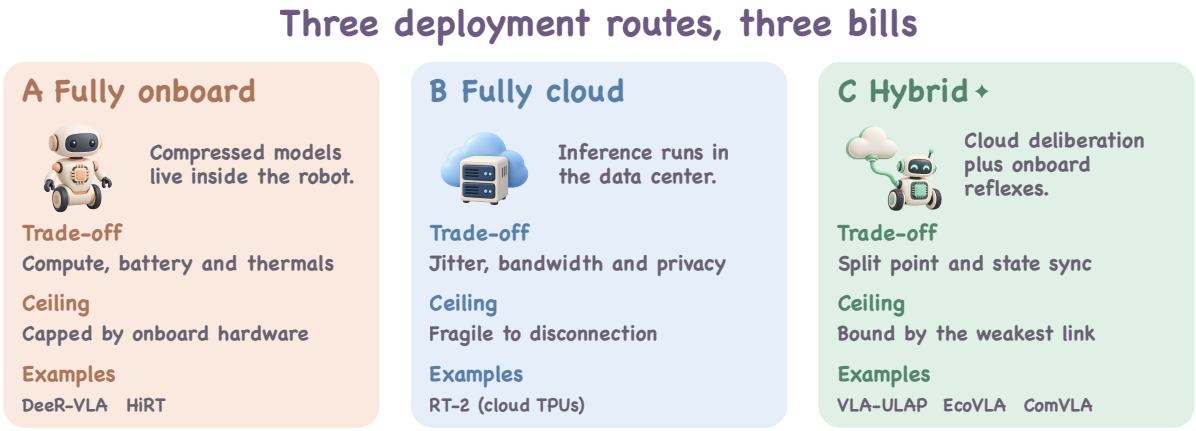


Figure 9 Three deployment routes and their cost profiles.

The Dynamic Splitting Problem. Hybrid deployment must decide, per control step, whether computation remains onboard or is offloaded. Split computing has a long tradition in mobile systems [186]; the robotic setting contributes a new constraint set—hard real-time service-level objectives (SLOs), energy budgets, and wireless bandwidth, all varying per step—which turns the split from a one-shot partition into an online scheduling decision. Three systems approach this decision from complementary perspectives.

1. Energy: EcoVLA [16] decides per step between onboard execution and offloading, trading energy against deadline, with edge-side agentic workloads sharpening the same trade-off [14].
2. Communication: the wireless link bounds the sensor payload per step, pushed to the extreme in 6G-connected robotics [13].
3. Serving: a single edge server multiplexes a fleet under per-robot SLOs, building on PagedAttention [204] and RadixAttention [445] from the LLM-serving toolkit [17]).

The synthesis is that the split point is not a static topology choice. Instead, it operates as a scheduling decision that adapts continuously to energy, bandwidth, and load. This dynamic characteristic defines the unique problem shape that distinguishes agentic infrastructure from classical cloud computing.

7.3 Agent Runtimes and Interfaces

Section 7.2 has discussed where models execute. This section introduces a complementary topic, how models are used. The runtime handles scheduling and integrates upward into the algorithm stack, while MCP handles connection and integrates downward into the hardware stack.

MCP as the southbound interface. MCP, introduced by Anthropic in 2024, is an open protocol that any large model can invoke any external tool or data source in one common format. It defines three capability classes, including *tools* (callable functions), *resources* (readable data), and *prompts* (reusable templates). An early robotics instantiation is the ROSBag MCP Server [7], which exposes robot recordings as MCP servers so that large models can query them in natural language. DriveMCP [15] applies the same recipe in the driving domain. MCP bears on infrastructure at two levels. At the interface level, it standardizes the agent's southbound surface, and runtimes cease to accumulate per-tool glue code. At the architecture level, once robot capabilities are wrapped as MCP tools, the robot becomes a set of discoverable, callable, and composable services, enabling fleet scheduling, skill markets, and cross-embodiment reuse.

Runtimes and the installed base. Above these interfaces, a supervisory layer manages subtask queueing, state transfer across models, failure recovery with degradation, and multi-robot resource allocation. Recent systems organize these functions into operating layers for embodied agents. For instance, RoboOS [5] supports cross-embodiment execution and multi-agent collaboration, while RoboOS-NeXT [6] provides a unified memory

substrate for lifelong learning and fleet-wide experience sharing. The Data Distribution Service transport in ROS 2 carries millisecond-rate control traffic with quality-of-service guarantees. However, ROS 2 was engineered to move messages between nodes rather than to manage intelligent agents. It lacks native concepts for tool discovery, memory substrates, and task-graph abstractions over heterogeneous policies. Conversely, the Model Context Protocol targets tool invocation at conversational timescales rather than high-frequency control streams. This creates a timescale mismatch where tool calls operate at the second scale while control streams operate at the millisecond scale. No single interface can serve both domains. A defensible architecture preserves the native interface of each layer, maintaining ROS 2 below and the protocol above, while locating the necessary translation logic within the runtime. This translation layer represents precisely the thinnest component in current systems.

Every single item addresses a systems concern. Infrastructure has evolved into a complex systems problem. The memory layer crowns the entire stack. Fleet-wide experience pools [6] and data factories [19] already feed offline training loops, leaving the primary missing operational counterpart as retrieval-grade responsiveness for what the robot should remember next.

7.4 Hardware and Embodiments

This subsection is deliberately brief, as it completes the stack while deferring depth to specialized hardware surveys. We categorize hardware systems into compute substrates, embodiment morphologies, and end-effector modalities.

Compute substrates. The device side is dominated by edge modules such as Jetson Orin and Thor, whereas the cloud side comprises massive GPU fleets and TPUs. Although robotics-specific silicon is continuously emerging, onboard compute capabilities grow far more slowly than model parameter sizes. Consequently, the deployment tension discussed in Section 7.2 remains structural rather than transitional. NVIDIA leads a vertically integrated path with systems like GR00T N1 [278], an open foundation model for humanoid robots featuring a dual-system architecture that matches the slow-deliberation and fast-reflex organization traced throughout this chapter, which ships directly alongside Jetson Thor hardware.

Embodiment morphologies. Morphologies span humanoids in bipedal and wheeled-legged configurations, quadrupeds, and mobile bases equipped with arms. Morphology choices fix both control frequencies and payload capacities, where wheeled-legged platforms favor endurance and stability while bipedal humanoids prioritize general mobility at a higher energy cost. This hardware constraint feeds back directly into the splitting policy because highly dynamic motions must be computed entirely onboard.

End-effector modalities. End-effectors include parallel grippers, multi-fingered dexterous hands, and specialized tactile sensors. These physical interfaces cap the overall skill library since the capabilities of a dexterous hand vastly exceed those of a standard gripper. However, the dedicated study of end-effectors belongs to a specialized hardware community outside the primary scope of this survey.

8 Safety and Trustworthiness

Agentic embodied systems connect perception, planning, execution, and feedback in a continuing loop. Beyond the action interface of a VLA policy, an agentic system can actively gather missing information, retain safety-relevant state, insert prerequisite actions, request assistance, and revise a plan after an execution error. These capabilities also introduce new failure modes: a planner proposes a risky instruction, a memory may preserve a stale or incorrect fact, a tool call may exceed the user's authority, and a policy may remain unsafe despite a convincing verbal explanation. This chapter therefore treats **Safety and Trustworthiness** as system-level properties of AEI, rather than properties of the language model or VLA in isolation.

We organize the chapter around the safety responsibilities created by this loop. Section 8.1 defines the protected entities, hazard types, temporal scope, and autonomy boundaries used throughout the discussion. Section 8.2 examines safety-critical perception: how an agent grounds risks in objects, relations, dynamics, history, and uncertainty. Section 8.3 then studies how a long-horizon task can be decomposed and reordered so that preconditions, mitigation actions, and termination obligations are satisfied. Section 8.4 covers policy-level

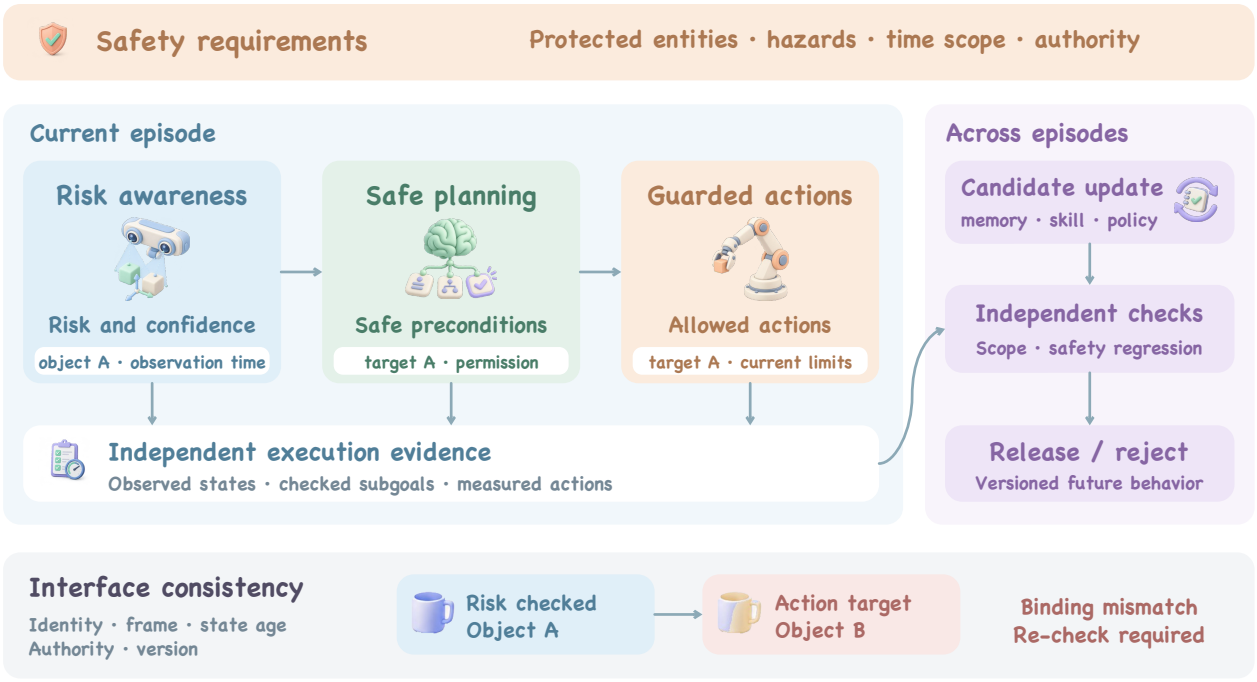


Figure 10 Safety contracts and evidence across agentic embodied intelligence. Shared requirements apply within an episode and to persistent updates across episodes. Risk judgments, planned subgoals, and action constraints must preserve their referents and assumptions. Execution evidence supports independent checks before an update is accepted. The lower example shows a broken object binding: a risk check for object A cannot justify an action on object B. Identity, coordinate frame, state age, authority, and version are complementary consistency requirements. This is a conceptual synthesis, not an implemented architecture or an end-to-end safety guarantee.

constraints and runtime enforcement at the action interface. Section 8.5 addresses reflection, memory, and updates across episodes. Finally, Section 8.6 asks what evidence is sufficient to support a safety claim and identifies the gaps that prevent current systems from providing end-to-end assurance. Figure 10 summarizes the constraints and evidence that must remain connected across these responsibilities; evaluation applies throughout, rather than only after execution.

8.1 Scope and Safety Requirements

Definition and Scope of Safety. For each episode, safety must account for the observations, actions, latent state, and memory that evolve over time. Four overlapping risk situations are especially important:

1. An adversary supplies a malicious instruction, scene perturbation, prompt injection, or poisoned memory;
2. A benign task contains a contextual hazard or a missing precondition;
3. The agent, tool, or controller deviates from the intended action;
4. An unresolved hazard persists after the nominal task has succeeded.

The fourth case is especially important for long-horizon agents: a task may satisfy its terminal predicate while leaving a person, appliance, or object in a dangerous state. Existing robot-safety surveys, human–robot collaboration studies, and safe reinforcement-learning surveys provide complementary taxonomies of these issues [70, 120, 134, 226, 351]. For generalist robots, context-dependent and delayed consequences extend the safety problem beyond geometric constraints alone [329].

These situations must be translated into requirements that can be checked during operation. A safety specification should state what is protected, which situation counts as a hazard, when the condition must

hold, and what authority the agent has. Examples include excessive contact force, entering a forbidden region, operating an appliance with an incompatible object, exposing private information, or executing an action without authorization. Safety must be checked throughout the episode and again at termination. The final check includes clean-up and hand-over obligations that a task-success label can miss, such as leaving an appliance in a safe state or returning a tool to an authorized person.

Autonomy, Utility, and Human Oversight. These requirements also delimit the agent's authority to act. A deployment should therefore specify which actions may be taken autonomously, which require confirmation, and which are prohibited. These boundaries should be enforced at the tool and controller interfaces rather than inferred from a generated explanation. Safety does not mean refusing every uncertain request. A useful system should complete benign tasks, mitigate contextual hazards when an authorized safe procedure exists, and refuse actions that are prohibited or cannot be executed within the declared limits.

8.2 Safety-Critical Perception and Risk Awareness

Agentic safety starts with a belief about what may go wrong. A risk module should combine current observations, object and relation estimates, action history, memory, and uncertainty. Risk must be grounded in physical states and relations (inside, supporting, touching, behind, or near), rather than in a free-form textual warning. The following two sources of evidence are complementary: the current scene and previously observed states or experiences.

Current-scene risk recognition illustrates the importance of this grounding. HomeGuard uses a VLM-based safeguard to identify contextual risks in household tasks [249]; its purpose is to recognize a dangerous relation or object state before the main policy acts. SafeRelBench evaluates spatial-relation-induced process hazards, including support, containment, and proximity, alongside non-spatial control cases [394]. The SAFEL evaluation framework and its EMBODYGUARD benchmark diagnose unsafe-command refusal and failures of safe, executable planning, rather than treating all failed episodes as equivalent [331]. These works complement object detection and collision checking: foil inside a microwave, a hand underneath a falling object, and a hot pan near a child illustrate how state and relations determine safety, not object presence alone.

Longer-horizon agents can use history to predict a hazard that is temporarily out of view. Safe task planning with graph memory represents persistent object, relation, and risk facts and uses them during replanning [225]; the just-in-time memory line retrieves and writes safety-relevant experience when a new situation resembles a previous failure [326]. The evidence supports a narrower claim than universal hazard discovery: memory helps when the relevant object, relation, or failure has been observed and represented. It cannot establish that an unobserved defect does not exist. Memory can also be stale, overgeneralized across embodiments, or poisoned, so every retrieved fact should carry provenance, confidence, time, and an applicability condition.

Current observations and memory may still leave a safety-critical condition unresolved. When uncertainty can change the pending decision, the agent should consider additional observation, a bounded probe, or clarification from the user. Moving the camera, opening a drawer, or asking a question can reveal missing information, but their value must be weighed against the time and physical risk of acquiring it; an informative probe is not necessarily safe. Uncertainty can trigger active observation instead of being silently converted into a confident plan. An agent's verbal explanation is evidence about its internal claim, not evidence that the executed trajectory was safe.

8.3 Safe Decomposition and Execution of Long-Horizon Tasks

The distinctive opportunity of an agentic system is to transform a safety requirement into a plan constraint and then continue the task after mitigation. A safe planner should prefer a useful plan only after checking its state, ordering, and authorization constraints and confirming that its predicted hazard exposure is acceptable. For example, the constraints may require that a cover is removed before heating, that a human remains outside a protected region during motion, or that the robot asks for confirmation before an irreversible action. This is different from filtering a final action: it accounts for hazards introduced by intermediate states and for obligations that remain after task completion.

Plug in the Safety Chip compiles natural-language safety requirements into constraints that can intervene



Figure 11 Safety-conditioned task responses. The same mug-transfer goal admits different responses under different conditions. Direct execution changes the task state. Removing an obstruction changes the world and requires rechecking before transfer. Observing an unobserved region changes the agent's information; in the illustrated case it reveals an obstacle without removing it. Declining an unauthorized transfer preserves the authority boundary and leaves the task unexecuted. The hand symbol denotes a permission restriction, not a physical obstacle. These illustrative cases may overlap and are not an exhaustive decision procedure. Mitigation and information gathering must themselves respect the safety requirements.

in an LLM-driven robot planner [406]. Generating Robot Constitutions explores a related route in which semantic principles and benchmark cases are made explicit for robot decision making [317]. These approaches make requirements inspectable; formal checks additionally require a suitable symbolic representation and verifier. The guarantee is only as strong as the specification, its translation, and the grounding of symbols in the physical scene. A formally verified plan can still be unsafe if "hot" was not detected, if the wrong object was bound to a symbol, or if an omitted constraint is the actual source of harm.

Given these constraints, the planner must choose a response rather than merely label a task unsafe. Satisfying them can require adding an action, changing an order, requesting confirmation, or terminating safely. A hazardous command is not the same as a benign command whose context makes one step dangerous: the latter may admit a safe continuation after mitigation. The mitigation itself must respect the same constraints, and its result must be checked before resuming the task. Figure 11 illustrates how known safety conditions, repairability, authorization, and missing information lead to different admissible responses to the same goal.

Selective escalation is another proactive behavior. When a planner is uncertain about a semantic precondition, KnowNo ("Robots That Ask for Help") uses conformal prediction sets to trigger human assistance [306]. Its statistical task-completion guarantees rely on the conformal calibration assumptions and the modeled planning and assistance setting; they do not certify perception, low-level execution, or physical safety during a delayed human response. Help-seeking should be evaluated jointly with response latency, user burden, and the probability that the human's answer is itself misinterpreted. Otherwise an agent can obtain a high safety score by asking indefinitely. The desired behavior is calibrated autonomy: act when the risk is low and reversible, seek information when it is valuable, and stop or hand over when the remaining risk is unacceptable.

8.4 Safe Policy and Runtime Enforcement

Planning constraints cannot replace an execution layer. The policy may produce an action outside the dynamics model, a VLA may emit a kinematically invalid command, or an unexpected contact may make the plan obsolete. Safety-critical execution therefore benefits from an independent monitor or safety filter that observes the state and can modify, delay, or stop the proposed action. We distinguish training-time constraints, predictive checks before committing actions, and intervention at the runtime action interface. These are intervention stages, not mutually exclusive method families.

Training-Time Safety Constraints. Physical human–robot safety has long combined collision measurements, compliant design, and speed and force limits; these constraints remain necessary when a learned policy is inserted into the loop [134, 351]. Constrained policy optimization and safe RL optimize return subject to safety-cost constraints [23]. SafeVLA integrates safety constraints into VLA alignment through constrained learning [429]. These methods can learn less conservative behavior than a fixed rule set, but their guarantees depend on cost labels, constraint observability, model accuracy, and the coverage of the training distribution. A low average cost does not rule out a rare, high-severity event.

Predictive Checks before Execution. Predictive safety filters and reachability analysis provide complementary ways to reject actions whose predicted future states leave the set [35, 352]. Here, the set is the declared set of admissible states; checking occurs before committing the next action or short rollout, not only once before deployment. Pre-VLA uses learned safety-confidence and advantage estimates to screen candidate action chunks and trigger resampling before physical execution or world-model imagination [336]. This learned assessment is distinct from a formal proof that a rollout remains safe. Model-based filters additionally require sufficiently accurate dynamics and sensing, a nonempty feasible set, and a clear definition of what the safe set protects.

Runtime Action-Interface Safeguards. Control barrier functions (CBFs) can support forward invariance of a specified safe set, under the required sensing, dynamics, and feasibility assumptions, and can project a nominal action onto a feasible action set [36]. Safety layers and barrier-based learning combine policy learning with action-level constraints [85, 95]. VLSA adds a plug-and-play constraint layer that can be attached to an existing VLA [146]. Latent safety filters use compact visual representations for action filtering [340]; uncertainty-aware filtering additionally targets out-of-distribution failures [315]. These filters are attractive as a last line of defense because they can run at a higher rate than a language planner. Predictive checking and runtime filtering can overlap: a runtime filter may verify every command immediately before it is issued.

A useful runtime contract includes at least an action validity predicate, a maximum force and speed, a workspace and human-distance bound, an authority token, a timeout, and an independent emergency stop. Its checks must refer to the same object identities, coordinate frames, and sufficiently recent state as the planner; stale or incompatible inputs can invalidate the protection. A filter that prevents a collision does not establish that the robot was authorized to open a cabinet or that a private object was not recorded. Results should distinguish empirical improvement from formal guarantees and state the assumptions under which the latter hold.

8.5 Safe Reflection, Memory, and Evolution

Agentic systems learn from their own trajectories. Reflection can identify that an action violated a precondition, memory can make that fact available in a later episode, and a policy update can change future behavior. Whereas Section 6 examines how experience improves capability, the concern here is whether a persistent change preserves safety requirements beyond the current episode. This perspective explains why agentic capability is both a safety opportunity and a new attack surface. It adds cross-time context, active mitigation, and calibrated help-seeking, but it also adds tool permissions, writable memory, self-generated plans, and update mechanisms.

A safety memory entry should record the evidence, affected embodiment and environment, validity interval, confidence, and the condition under which it may be retrieved. The system should prefer a narrow rule with provenance over an unsupported generalization such as "this object is always safe." Just-in-time memory and long-term graph memory provide early examples of turning a failure or risk observation into a future planning

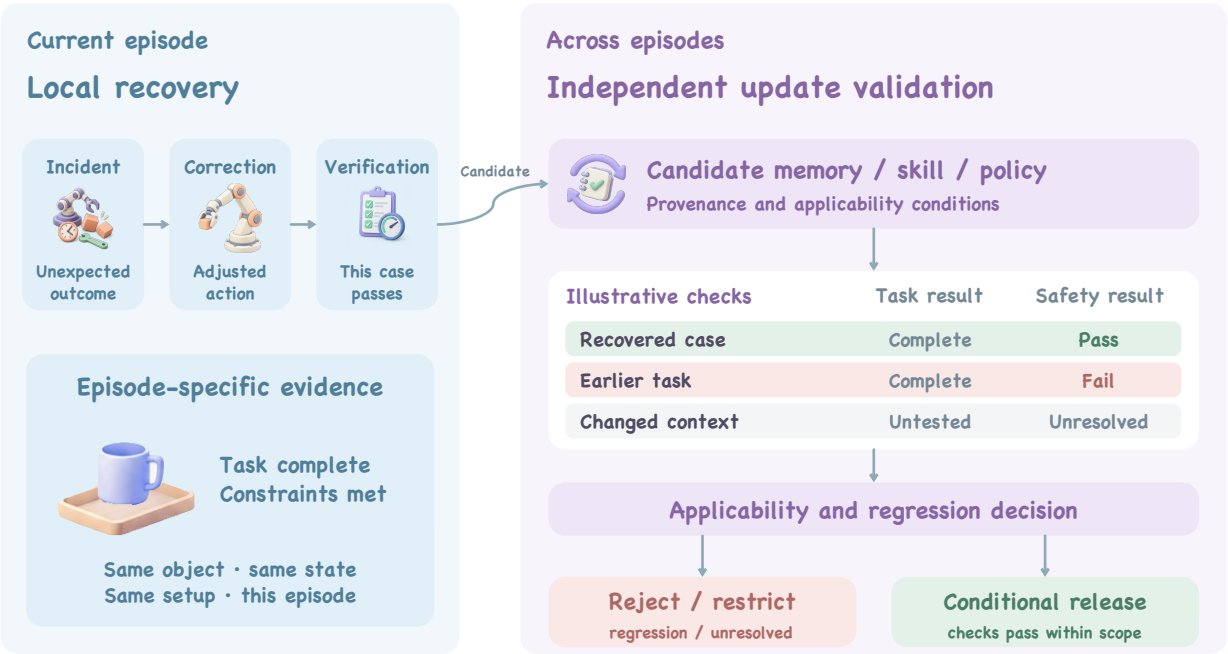


Figure 12 Safety validation of persistent updates. A verified local correction can motivate a candidate update but does not establish its safe reuse. The illustrative checks contrast task completion with safety outcomes: an update may reproduce the recovered case yet violate a constraint on an earlier task or remain unevaluated in a changed context. Such evidence calls for rejection, revision, or restriction to an independently validated scope, not unconditional release. Conditional release requires the declared checks to pass and retains applicability limits, versioning, monitoring, and rollback. The displayed outcomes are schematic, not experimental results; software rollback cannot undo physical harm.

signal [225, 326]. Reflection is useful when it changes an executable prerequisite, skill, or monitor; a post-hoc natural-language summary without a retrieval or policy effect is not evidence of learning.

Beyond individual memory entries, persistent skill, policy, and system changes require a separate acceptance decision. A safety-specific outer loop should therefore proceed in plain stages: analyze incidents and near misses, propose a candidate memory, skill, or policy update, replay earlier cases and run red-team tests, and only then register a new version. Promotion additionally requires testing benign and hazardous counterfactuals, checking regression on old tasks, and retaining an automatic rollback path. The update must be versioned and reversible. Figure 12 contrasts episode-specific verification with the independent evidence needed to qualify a persistent update; the depicted test outcomes are illustrative. Evidence used to accept an update should be independent of the mechanism that proposes it.

Safe evolution must also test for memory staleness, cross-embodiment transfer errors, data poisoning, and catastrophic forgetting. Passing earlier tests does not establish safety under all future conditions, and restoring a software version cannot undo physical damage already caused. The central assurance question is therefore not whether an agent has a safety module, but whether every new capability has a bounded authority, an observable failure signal, and a recovery path.

8.6 Evaluation and Assurance Gaps

We organize safety evidence as a claim–assumption–test triple, informed by the diagnostic emphasis of recent embodied-safety benchmarks and surveys [226, 248, 331, 410]. The claim might be "the agent removes a contextual hazard and finishes the task"; assumptions include the correctness of the scene representation, the action interface, and the hazard label; and the test must expose the assumptions through perturbations and independent observation. Table 9 summarizes the evidence and limits of representative benchmarks and

Table 9 Representative safety evaluation benchmarks.

Evaluation	Useful evidence	Remaining boundary
Robot constitutions and semantic cases [317]	Whether stated principles and semantic hazards are recognized and followed	Does not test noisy contact dynamics or long-horizon execution
AGENTSAFE [410]	Perception–planning–execution behavior under hazardous instructions	Does not cover all benign tasks requiring active mitigation
IS-Bench [248]	Interactive, prerequisite, and postcondition safety in household tasks	Safety remains relative to its task ontology and simulator/robot setup
SafeRelBench [394]	Process risks caused by support, containment, and proximity relations	Limited coverage of hidden state, multi-robot interaction, and hardware faults
SafeVLA [429], VLSA [146], latent filters [340]	Policy safety costs and action-level constraint enforcement	Does not by itself establish privacy, authorization, or persistent-memory safety

safeguard evaluations.

For benign tasks, report task success, safe completion (the whole trajectory and terminal state satisfy the declared predicates), and erroneous refusal. For repairable contextual hazards, report whether preconditions are established before exposure, whether required mitigation is attempted, mitigation success, exposure duration and near misses during mitigation, safe completion afterward, and residual risk at hand-over. For prohibited or irreparable actions, report correct refusal or safe termination separately. IS-Bench evaluates interactive safety in daily household tasks and includes dynamic, prerequisite, and postcondition failures [248]. AGENTSAFE evaluates embodied agents on hazardous instructions and is useful for diagnosing instruction-level refusal and execution failures [410]. The two types of benchmark answer different questions: a hazardous command is not the same as a benign command whose context makes one step dangerous. Separating these outcomes prevents blanket refusal from concealing an inability to mitigate repairable risk.

We recommend reporting metrics with explicit denominators. Safety should never be computed only over successful episodes, because that conditions away the failures that matter. Likewise, mitigation measured only on detected hazards must be distinguished from end-to-end performance over all episodes containing the hazard. Awareness measures include risk recall, false alarms, localization, relation and temporal-order accuracy, calibration, and detection latency. Execution measures include violation frequency and severity, minimum safety margin, intervention rate and necessity, response latency, and task delay. Safety Gym illustrates why task return and safety cost should be reported separately [302]. Protocols should balance hazardous and non-hazardous controls, stratify by hazard type, scene, duration, and embodiment, and publish the safety predicates and trigger times used by the evaluator [302, 317, 394]. Test episodes should include occlusion, out-of-view hazards, changed object states, misleading descriptions, and positive and negative relation controls. Environment state and trajectory logs are preferable to an LLM judge; if an LLM judge is used for semantic labels, it requires human spot checks and inter-rater agreement. Repeated trials should include uncertainty estimates.

The strongest comparison is a controlled ablation with the same backbone, action policy, observation budget, tools, and compute: a baseline policy, static safeguards, risk reasoning, memory, and active mitigation. This follows the component-level diagnosis encouraged by EMBODYGUARD and the safety-filter literature [315, 331, 340]. It should include clean and adversarial scenes, out-of-distribution objects, memory corruption, delayed sensing, and updates followed by safety regression tests. This prevents attributing a gain from a larger model or more tool calls to agentic safety itself.

Three gaps remain especially important. First, specifications for hidden state, irreversible consequences, near misses, and prolonged exposure are not yet consistent across benchmarks [226, 331, 394]. Second, safety of individual modules does not compose automatically: planner, memory, policy, and controller may share a failure mode or make incompatible interface assumptions [36, 70]. Third, active safety evaluation should reward discovering and removing hazards while accounting for unnecessary refusal, delay, repeated help requests, and risks introduced by mitigation itself [248, 306]. A finite set of accident-free trials is evidence for the tested setting, not a proof that the probability of harm is zero. Progress on agentic embodied safety will

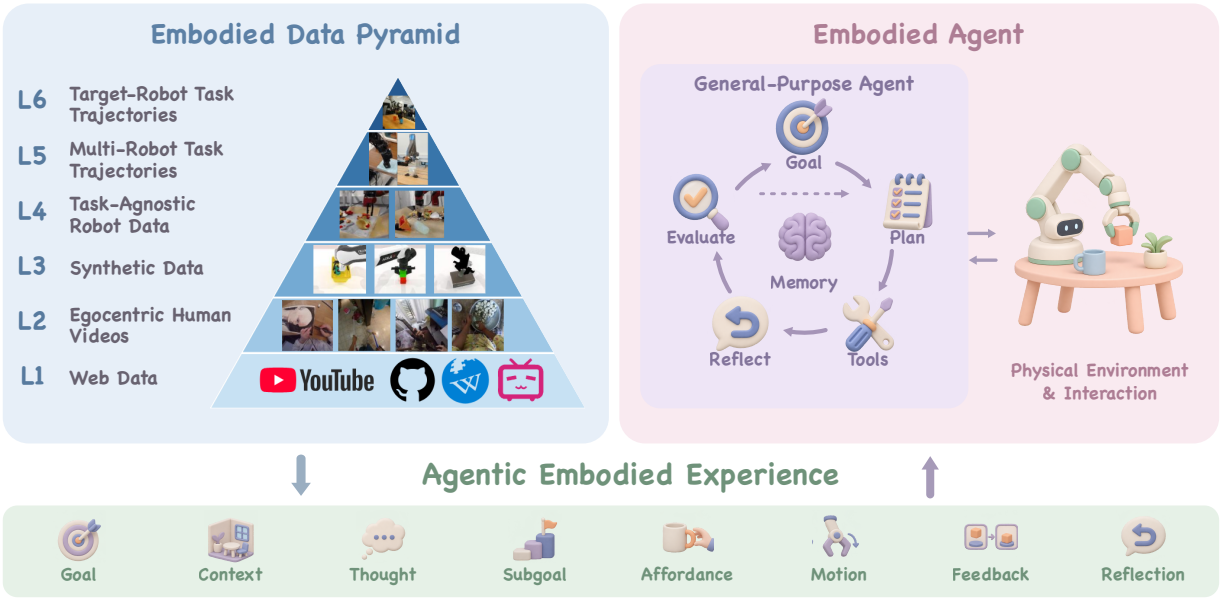


Figure 13 From the embodied data pyramid to Agentic Embodied Experience. The six-level data pyramid of Motus [57] organizes complementary sources, ranging from web data and human videos to synthetic data, task-agnostic robot data, and multi-robot and target-robot task trajectories. To support embodied agents that integrate goal-directed planning, tool use, evaluation, and reflection with physical interaction, these resources must connect decisions to execution, observed consequences, and subsequent revision. The Agentic Embodied Experience schema captures these connections through eight linked elements: goal, context, thought, subgoal, affordance, motion, feedback, and reflection, organizing heterogeneous data into experience that supports adaptive embodied behavior.

depend on this evidence chain as much as on stronger models.

9 Data Ecosystem

Agentic embodied intelligence requires data that connect task goals and decision rationale to physical execution, observed consequences, and subsequent correction. Such experience supports long-horizon behavior, failure recovery, and adaptation as tasks and environments change. This chapter examines the data ecosystem through the lens of *Agentic Embodied Experience*, focusing on how experience is organized, generated, and used for continued learning. We first introduce the embodied data pyramid and an eight-field experience schema linking decisions to interaction outcomes. We then examine four complementary generation routes: active exploration, human videos, simulation, and human corrections. Finally, we connect task selection, execution evaluation, and policy improvement in an agentic data flywheel, before discussing the limitations and data requirements of sustained autonomous operation.

9.1 From the Data Pyramid to Agentic Embodied Experience

Data Pyramid. The data pyramid is a widely used view of embodied data organization. Here we adopt the six-level formulation of Motus [57], which organizes resources by source and training role relative to a target robot. From bottom to top, its levels are:

- **L1: Web Data.** General text, images, and videos supply visual, language, and world-knowledge priors.
- **L2: Egocentric Human Videos.** Human activity recordings provide interaction, motion, and task knowledge.
- **L3: Synthetic Data.** Simulation and digital twins provide controllable, scalable scenes and interaction data.

- **L4: Task-Agnostic Data.** Robot sampling covers motion and interaction spaces without requiring semantic task objectives.
- **L5: Multi-Robot Task Trajectory Data.** Aggregated task demonstrations support knowledge sharing across embodiments.
- **L6: Target-Robot Task Trajectory Data.** Demonstrations for the target platform support task learning and adaptation.

The pyramid captures a broad shift from abundant general priors to targeted robot experience. For agentic learning, a further question is whether the data connect a decision to its actual execution, observed consequences, and subsequent revision. Figure 13 illustrates how the data pyramid and embodied agent requirements motivate this experience view.

Agentic Embodied Experience Schema. General-purpose agents pursue goals through a recurring cycle of deciding, acting, observing outcomes, and revising subsequent behavior. ReAct [408] illustrates the interplay of reasoning and action, while Reflexion [321] shows how feedback can inform later attempts. For embodied agents, this cycle must also connect decisions to interaction targets, executed motion, and physical consequences. We therefore introduce the following schema for organizing *Agentic Embodied Experience*:

$$\mathcal{E}_{\text{agentic}} = \langle \mathcal{G}_{\text{goal}}, \mathcal{C}_{\text{context}}, \mathcal{T}_{\text{thought}}, \mathcal{Z}_{\text{subgoal}}, \mathcal{A}_{\text{affordance}}, \mathbf{a}_{\text{motion}}, \mathcal{O}_{\text{feedback}}, \mathcal{R}_{\text{reflection}} \rangle. \tag{6}$$

- **Goal ($\mathcal{G}$).** An explicit task objective or specification establishes the intended outcome.
- **Context ($\mathcal{C}$).** Visual observations, states, and relevant history ground decisions in the current situation.
- **Thought ($\mathcal{T}$).** Annotated or generated explanatory rationales link available evidence to decision choices.
- **Subgoal ($\mathcal{Z}$).** Plan steps or temporally annotated action/skill segments organize execution into stages.
- **Affordance ($\mathcal{A}$).** Contact/interaction annotations, and feasibility labels ground where and how to interact, with different degrees of physical specificity.
- **Motion ($\mathbf{a}$).** Robot state trajectories, control actions or targets, and human hand poses or position sequences record the embodied realization of behavior.
- **Feedback ($\mathcal{O}$).** Interaction effects in temporal or before–after observations provide evidence of progress and deviation, including passively observed physical events.
- **Reflection ($\mathcal{R}$).** Post-execution diagnosis or corrective commentary supports behavioral revision and later attempts.

Existing Resources. Table 10 maps representative resources to both the pyramid and the experience fields. These are complementary axes: source and training role versus recorded supervision. Target-robot demonstrations often provide goals, observations, motion, and outcomes. ECoT [422], RH20T-P [83], and AHA [107] enrich experience with reasoning, skill segmentation, and failure diagnosis, respectively. Human video can also supply rich decision and diagnostic supervision. Ego-Exo4D [127] covers all eight fields across annotated subsets, but this resource-level coverage does not mean that they co-occur in one sample or form a unified, temporally aligned schema. The remaining challenge is to connect these complementary signals into experience records usable for agentic learning.

Failure data are particularly valuable for connecting feedback to revision. RoboMIND's failure subset [380], RoboFail [247], and AHA [107] pair unsuccessful executions with diagnostic annotations, exposing mismatches between intended and observed effects. Within the schema, failed executions provide direct feedback ($\mathcal{O}$) that grounds reflection ($\mathcal{R}$), supporting error detection, diagnosis, and corrective action selection. REFLECT [247] illustrates how execution histories support failure explanation and correction planning. Preserving subsequent recovery attempts and their outcomes further connects diagnoses to tested revisions, providing supervision for adaptive behavior when initial plans break down.

Table 10 Data resources across the embodied data pyramid, ordered from L1 (bottom) to L6 (top), and their explicit coverage of Agentic Embodied Experience fields. A checkmark indicates that the named resource or specified subset provides the field; a cross indicates that it is not explicitly supplied under the definitions above.

Dataset / data resource	Level	$\mathcal{G}$	$\mathcal{C}$	$\mathcal{T}$	$\mathcal{Z}$	$\mathcal{A}$	$\mathbf{a}$	$\mathcal{O}$	$\mathcal{R}$
LAION-5B [313]	L1	×	✓	×	×	×	×	×	×
WebVid [49]		×	✓	×	×	×	×	×	×
Ego4D [126]	L2	×	✓	×	✓	✓	✓	✓	×
EPIC-KITCHENS-100 [96]		×	✓	×	✓	✓	×	✓	×
Ego-Exo4D (annotated subsets) [127]		✓	✓	✓	✓	✓	✓	✓	✓
EgoDex [141]		✓	✓	×	×	×	✓	✓	×
OpenEgo [167]		✓	✓	×	✓	×	✓	✓	×
EgoRe-5M (CoT / temporal QA / grounding subsets) [285]		×	✓	✓	✓	✓	×	✓	×
EPIC fine-grained affordance annotations [415]		×	✓	×	✓	✓	×	✓	×
TextAFF80K [412]		✓	✓	×	×	✓	×	×	×
VSAD [239]		×	✓	×	×	✓	×	✓	×
Causal Triplet (EPIC-KITCHENS subset) [245]		×	✓	×	×	×	×	✓	×
RoboVQA (human video–QA subset) [316]		✓	✓	×	✓	✓	×	✓	×
Causal Triplet (ProcTHOR subset) [245]	L3	×	✓	×	×	×	×	✓	×
CLEVRER [409]		×	✓	×	×	×	×	✓	×
CausalWorld [26]		✓	✓	×	×	×	✓	✓	×
RoboTwin [273]		✓	✓	×	×	×	✓	✓	×
robomimic (simulation subset) [264]		✓	✓	×	×	×	✓	✓	×
AHA [107]		✓	✓	×	✓	×	×	✓	✓
RoboFail (simulation subset) [247]		✓	✓	×	✓	×	✓	✓	✓
BAIR robot pushing [108]	L4	×	✓	×	×	×	✓	✓	×
RoboNet [98]		×	✓	×	×	×	✓	✓	×
AnyPos [341]		×	✓	×	×	×	✓	×	×
RH20T [109]	L5	✓	✓	×	×	×	✓	✓	×
Open X-Embodiment [280]		✓	✓	×	×	×	✓	✓	×
RoboMIND [380]		✓	✓	×	×	×	✓	✓	×
RoboMIND (fine-grained language subset) [380]		✓	✓	×	✓	×	✓	✓	×
RoboMIND (failure subset) [380]		✓	✓	×	×	×	✓	✓	✓
RH20T-P [83]		✓	✓	×	✓	×	✓	✓	×
ShareRobot [169]		✓	✓	×	✓	✓	×	✓	×
RoboTurk [262]	L6	✓	✓	×	×	×	✓	✓	×
robomimic (real subset) [264]		✓	✓	×	×	×	✓	✓	×
ALOHA [442]		✓	✓	×	×	×	✓	✓	×
Mobile ALOHA [118]		✓	✓	×	×	×	✓	✓	×
BridgeData V2 [353]		✓	✓	×	×	×	✓	✓	×
DROID [191]		✓	✓	×	×	×	✓	✓	×
ECoT [422]		✓	✓	✓	✓	✓	✓	✓	×
RoboVQA (robot video–QA subset) [316]		✓	✓	×	✓	✓	×	✓	×
RoboFail (real-robot subset) [247]		✓	✓	×	✓	×	✓	✓	✓

9.2 Generating Agentic Embodied Experience

The Agentic Embodied Experience schema identifies the information needed to connect decisions, physical execution, and revision. We now examine how this information is acquired through active exploration, human videos, simulation, and human corrections. These routes enrich existing resources through new interactions and annotations, contributing complementary evidence about goals, action choices, and outcomes. Table 11 summarizes their contributions.

Collecting Experience through Active Exploration. Active exploration acquires experience where an agent's knowledge is incomplete. Interactive perception [62] treats actions as queries that reveal properties unavailable through passive observation. VIME [143], Self-Supervised Exploration via Disagreement [284], and Plan2Explore [314] use information gain or predictive disagreement to direct collection toward poorly understood transitions. New observations update the model and influence subsequent probes. A conceptual acquisition objective is

$$a_t^\star = \arg\max_{a \in \mathcal{U}_t} I(\Theta; o_{t+1} \mid h_t, a)\,, \tag{7}$$

where h_t is the interaction history, Θ denotes unknown environment properties or dynamics, $\mathcal{U}_t$ is the admissible action set, and I measures expected information gain. The target is epistemic uncertainty: knowledge gaps that additional interaction can reduce.

Uncertainty-guided interaction yields evidence about dynamics, geometry, and contact. Active exploration of object dynamics [312] uses information-seeking pushes to improve a real-robot dynamics model; active vision and touch [87] combine RGB-D views and touches to refine uncertain object surfaces. TacEx [161] targets tactile prediction uncertainty, collecting visual, force-map, and action transitions in simulation. Its exploration buffer is relabeled for offline pick-and-place learning, illustrating how contact-rich experience can support later tasks.

Autonomous collection also supplies broader interaction corpora. Learning to Poke by Poking [25] records pokes and their visual effects, while RoboNet [98] aggregates cross-robot trajectories collected using predefined random action distributions for dynamics and visual prediction. For evaluation, Explore until Confident [305] introduces HM-EQA, an HM3D-based question–answer dataset that tests whether visual exploration gathers sufficient evidence before answering. Across these settings, useful experience links the knowledge gap, selected probe, preceding observations, and measured outcome, retaining failed attempts alongside informative contacts.

Learning from Human Videos. Human activity videos provide experience across diverse objects, environments, and tasks. Ego4D [126] supplies everyday egocentric recordings with narrations and selected hand–object, state-change, and forecasting annotations; EPIC-KITCHENS-100 [96] provides long, unscripted kitchen videos with densely timed action segments. These sequences capture how people encounter, handle, and transform objects.

Attention and language annotations enrich these observations. Ego-Exo4D [127] synchronizes eye gaze, first- and third-person views, and time-indexed activity descriptions. Audio-visual gaze anticipation [206] predicts future fixations from Ego4D and Aria recordings, providing additional attention cues. OpenEgo [167] harmonizes hand poses and timestamped, intention-aligned action primitives across egocentric sources, while EgoRe-5M [285] adds QA pairs, generated stepwise rationales, and hand–object and temporal grounding. Together, these annotations connect observed activity to relevant objects, task stages, and explanations of action choices.

Interaction annotations further ground where and how to act. EPIC fine-grained affordance annotations [415] distinguish action, grasp, and object–object mechanical interaction categories; TextAFF80K [412] derives instruction-linked contact points and trajectories from human videos; VSAD [239] pairs egocentric interactions with 3D scene affordance regions. VRB [47] demonstrates their practical value by learning interaction locations and motions from human video and transferring these priors to robot exploration. Human videos thus contribute experience linking activity context, task steps, interaction regions, and observed motion and outcomes.

Generating Experience in Simulation. Interactive scene construction provides environments for generating experience. SimRecon [384] and ReplicateAnyScene [102] reconstruct simulation scenes from real-world

Table 11 Complementary routes to generating Agentic Embodied Experience.

Source	Primary signal	Strength
Active exploration	Physical responses to selected actions	Direct interaction feedback
Human videos	Intent, attention, and affordance annotations	Broad task diversity
Simulation	Plans, critiques, and tested revisions	Controllable and scalable
Human corrections	Language feedback and corrective actions	Targeted supervision

videos, while PhySGA [252] uses agent planning, physical simulation, and critic feedback to construct object compositions. These routes supply reconstructed or agent-constructed settings in which actions and corrections can be tested. GenSim [360] expands task programs and expert demonstrations; RoboGen [370] generates scenes, subtasks, training supervision, and skill demonstrations through a propose–generate–learn cycle.

Simulation also supports recording how plans change during execution. MALMM [327] separates planning, code generation, and supervision, using execution observations to trigger replanning. ReAd [436] learns an advantage critic from planned interactions and feeds its assessments back to a multi-agent planner; LogicGuard [124] critiques executed trajectories and proposes temporal constraints to avoid repeated failures. MADRA [358] adds multi-agent discussion of household task risks, followed by simulated execution and feedback-driven revision. These exchanges provide candidate plans, critiques, executable actions, and revised decisions grounded in simulator observations.

Preserving these exchanges would extend synthetic demonstrations into experience records linking initial states and goals to plans, tool calls, failed branches, tested revisions, and final outcomes. DeDer [88] provides a concrete annotation route by adding LLM-generated, self-checked rationales to embodied expert transitions for policy distillation. Recording similar decision–execution–revision links throughout simulated rollouts would supply supervision for both action selection and adaptation after failure.

Learning from Human Corrections. Human intervention adds targeted experience through language and action corrections. *Language correction* expresses changes to goals, action order, or local motion. Inner Monologue [156] incorporates human language feedback into planning; LILAC [93] adapts a shared-autonomy control space from utterances such as "to the right." Yell At Your Robot [319] collects occasional verbal corrections to improve a high-level policy across iterations, while DROC [428] distills correction sequences into knowledge retrievable for new tasks or objects. These records connect an utterance to the observed situation, the agent's interpretation, and revised behavior.

Action correction changes motion through takeover or physical guidance. DAgger [308] motivates expert supervision at learner-visited states; HG-DAgger [189] enables human takeover when needed. Intervention-weighted regression [263] uses corrective teleoperation segments, and RaC [148] records rewind-and-retry trajectories for recovery learning in long-horizon tasks. IntervenGen [142] expands a small number of human interventions into corrective trajectories across simulated states, while CR-DAgger [390] collects gentle delta-action corrections through compliant control as the robot continues operating. These approaches concentrate supervision near policy failures.

Physical guidance can additionally generate language supervision. Don't Yell at Your Robot [430] infers revised motion intentions from physical corrections and expresses them in language for later planning. Talk-and-Tweak [178] uses rule-based mappings to pair observed states and human corrective actions with language refinement commands. Retaining the original behavior, correction signal, and subsequent outcome makes such interventions reusable examples of how feedback leads to revised decisions and execution.

9.3 Closing the Loop: The Agentic Data Flywheel

The preceding section describes how Agentic Embodied Experience is generated. Turning this experience into sustained capability growth requires connecting task selection, execution evaluation, and policy improvement. The agentic data flywheel links these stages in a recurring cycle: curricula select informative tasks, execution produces new experience, evaluators add feedback, and learning converts that experience into improved

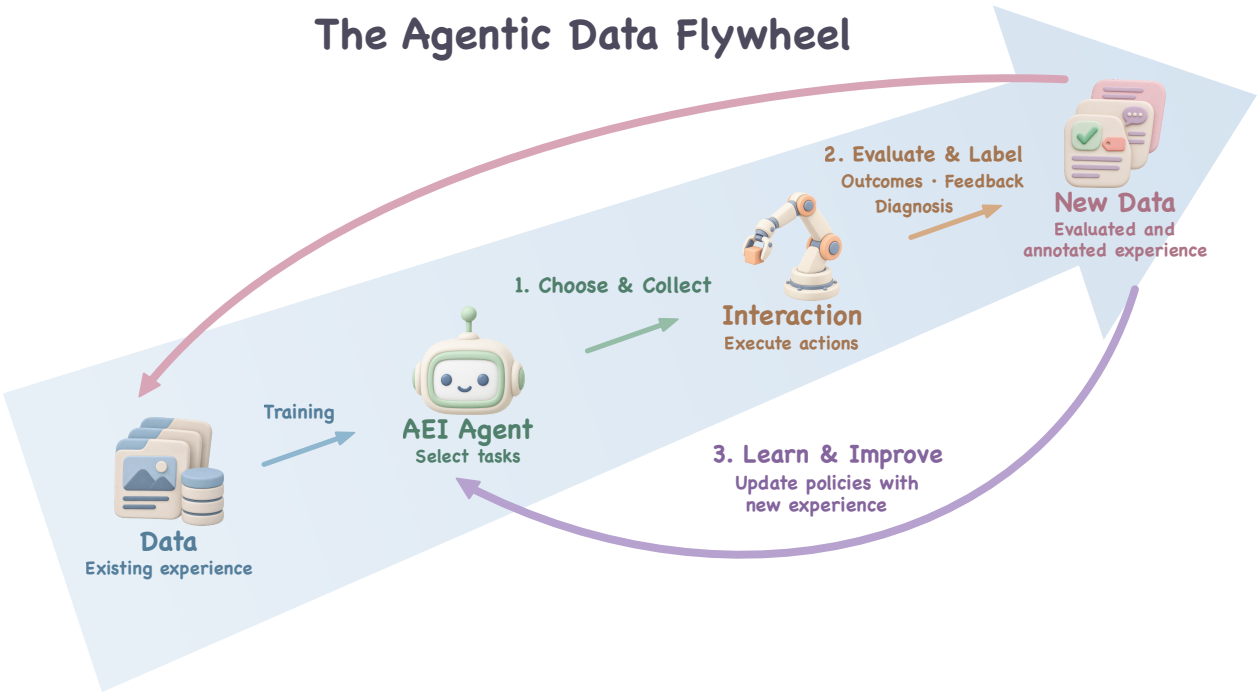


Figure 14 The agentic data flywheel. Agents trained on existing experience select tasks and collect interactions. Assessing interaction outcomes produces feedback, diagnoses, and annotated experience for policy updates. Improved agents then drive the next round of collection, linking data growth to capability improvement.

decisions and skills. Updated agents then shape the next round of collection. Figure 14 illustrates this cycle. Each iteration changes both what the agent can do and what data it needs next.

Choosing What to Collect Next. Autonomous curricula determine which interactions are collected as the learner's competence changes. GoalGAN [115] targets intermediate-difficulty goals, while ALP-GMM [290] samples parameterized environments according to estimated learning progress. Both direct experience acquisition toward tasks suited to current capabilities. GenSim [360] connects task generation to simulation demonstrations through goal-directed curricula or exploratory task expansion; AutoRT [30] proposes instructions for diverse real-world collection using teleoperation and autonomous robot policies. These approaches expand task coverage and connect proposed goals to new execution records. Within the flywheel, verified successes, failures, and recovery attempts reveal gaps in existing experience and guide subsequent task selection. Collection can thus progress from familiar tasks toward new conditions and behaviors, including situations that require correction or recovery.

Assessing Interaction Quality and Generating Feedback. Assessing the quality and outcomes of collected interactions produces success labels, preferences, and diagnostic feedback, turning execution records into labeled training data. SuccessVQA [105] formulates success detection as visual question answering using human reward annotations, enabling learned evaluators to assess new observations. RL-VLM-F [369] obtains task-conditioned preferences over image pairs from a vision–language model and learns a reward function from these labels, providing an embodied instance of reinforcement learning from AI feedback (RLAIF). AHA [107] uses procedurally perturbed demonstrations with failure annotations to learn diagnostic feedback, while Eureka [254] generates reward code for scoring simulator interactions. Together, these approaches supply outcome judgments, preferences, diagnoses, and reward values. Associated with execution observations, these signals provide supervision for policy learning and recovery, and identify experience gaps for the next curriculum.

Learning from Experience to Improve Policies. Collected experience supports improvement through retained

reflections, decision learning, and motor-policy training. Reflexion [321] stores feedback-derived verbal reflections that guide later attempts. DeDer [88] constructs self-checked rationale data from embodied expert transitions and distills them into smaller reasoning and planning policies. At the motor level, guided policy search [214] uses optimized trajectories to supervise visuomotor policies, while RoboCat [63] adds self-generated, action-labeled robot experience to subsequent training iterations. These approaches illustrate how reflections, rationale annotations, and executed trajectories contribute to better decisions and control. Linking reasoning and reflection to observations and validated actions makes deliberative experience available as supervision for more efficient policies. Improved agents can then attempt new tasks and collect experience under more demanding conditions; their successes and remaining failures inform the next curriculum, closing the loop between experience generation and capability growth.

10 Simulators, Benchmarks and Evaluation Protocols

Evaluating Agentic Embodied Intelligence (AEI) requires more than testing whether a robot can complete a fixed task. Classical embodied benchmarks enabled reproducible and comparable evaluation by standardizing environments, action spaces, and task success criteria. Platforms such as AI2-THOR [201], Habitat [309], RLBench [163], and Habitat 2.0 [339] established reproducible testbeds for navigation, manipulation, and household rearrangement. These benchmarks remain important, but they usually evaluate whether an agent completes a predefined task, such as placing a cup inside a drawer or navigating to a sofa. This is not sufficient for AEI, where successful task execution often depends on how the agent handles uncertainty and unexpected changes during interaction. Beyond completing the assigned goal, the agent may need to clarify ambiguous instructions, gather missing information, use or create tools, maintain task state over long horizons, detect failures, and revise its plan accordingly. Therefore, AEI evaluation should assess not only the final task outcome, but also the agent's behavior throughout the interaction process.

This chapter organizes AEI evaluation into three connected layers. First, a simulator defines the world in which agentic behavior can be exercised. For AEI, this world must expose not only geometry and rigid-body motion but also contact, multimodal sensing, persistent object states, and long-horizon state transitions. Second, a benchmark defines the capability being challenged. We therefore organize tasks by their cognitive and physical demands rather than only by surface categories such as navigation or grasping. Third, an evaluation protocol determines how the agent's performance is measured. It should evaluate not only whether the task is completed, but also the quality of the agent's reasoning, its ability to recover from failures, the physical validity of its actions, and its safety. It should also examine whether the results obtained in simulation are consistent with real-world robot performance.

The three layers should be designed together. First, the simulator must contain the physical properties needed by the benchmark. For example, a benchmark cannot test whether an agent knows that a sponge can absorb liquid if the simulator does not model that property. Second, the benchmark must expose the capability we want to measure. Even in a rich simulator, a task that only checks final success cannot tell whether the agent planned well, recovered from mistakes, or simply found a shortcut. Third, the metric should reward behavior that remains meaningful on real robots. A policy should not receive a high score only because it exploits a simulator-specific artifact that would not work in the real world. We therefore view AEI evaluation as an alignment problem among *world expressiveness*, *capability coverage*, and *measurement validity*.

10.1 Physics–Semantic Hybrid Simulation Foundations

From physics engines to cognitive-physical sandboxes. A conventional simulator mainly approximates a transition function from the current physical state and action to the next physical state. This abstraction is sufficient for many control problems, but it is too narrow for long-horizon embodied agents. AEI requires a simulator to preserve important object states even when the objects move out of view, to reflect the physical effects of the agent's actions, and to represent changes such as whether an object is open, filled, dirty, or broken. A useful abstraction is

$$(x_{t+1}, z_{t+1}) = F(x_t, z_t, a_t; \theta), \tag{8}$$

where x_t represents the physical state of the environment, such as object positions, orientations, velocities, and joint configurations. z_t represents higher-level task states and relations, such as whether a door is open,

whether a container is filled, or whether one object is inside another. a_t is the agent action, and θ contains physical properties such as mass, friction, and material parameters. Both x_t and z_t are needed because many everyday tasks involve more than changes in geometry. For example, opening a door changes its joint configuration as well as its semantic state from closed to open. Similarly, cleaning an object may change its state from dirty to clean even if its position remains almost unchanged. An AEI simulator should therefore model both physical dynamics and persistent task-relevant states, so that agents can observe the effects of their actions, track how the environment changes, detect failures, and verify task completion.

Contact dynamics and heterogeneous physical processes. The first requirement is a sufficiently expressive physical substrate. Rigid bodies and articulated mechanisms remain the base case because most manipulation depends on collision, friction, joint limits, and stable contact. Recent systems have substantially improved the scale at which such interactions can be simulated. ManiSkill3 [343] couples GPU-parallel simulation and rendering and reports more than 30,000 frames per second in some benchmarked environments, while covering mobile manipulation, dexterous control, humanoids, and other domains. MuJoCo Playground [421] similarly integrates MJX physics, batch rendering, training environments, and sim-to-real deployment in a lightweight stack. Isaac Lab [269] extends GPU-native simulation toward multi-modal robot learning with photorealistic rendering, actuator models, multi-frequency sensors, data pipelines, and domain randomization. These systems reduce the computational cost of interactive training and make large-scale perturbation studies practical.

However, rigid-body fidelity does not cover the full set of interactions implied by everyday autonomy. Clothing, cables, food, fluids, thin objects, and soft materials introduce deformation, topology changes, and contact regimes that are poorly approximated by rigid proxies. Recent work makes this limitation explicit. DLO-Lab [74] introduces a differentiable simulator for deformable linear objects such as ropes, cables, and rubber bands, modeling extensibility, elasticity, bending plasticity, and complex contact. SoMA [152] learns a real-to-sim neural simulator for soft-body manipulation and targets long-horizon tasks such as cloth folding. FlatLab [453] provides high-fidelity simulation and standardized evaluation for both rigid and deformable flat objects. These efforts point to a broader requirement: AEI simulation should support the physical processes that change task feasibility, not only those that are easy to compute. This includes deformable contact, fluid interaction, damage, and eventually thermal or material transitions when they are part of the task semantics.

Multimodal sensing as a channel for active physical inquiry. Visual observations alone are often insufficient for understanding physical interactions. AEI agents may also need depth, proprioception, force–torque signals, and tactile feedback to determine whether an interaction is physically successful. Taccel [233] addresses this gap for vision-based tactile sensing by simulating robot, sensor, and object interactions at high parallel throughput while preserving physically meaningful tactile signals. This capability is important not only for low-level control. It enables *epistemic actions*: actions whose immediate purpose is to reduce uncertainty. An agent may push an object to test friction, lift it to infer weight, touch a surface to localize contact, or perturb a container to infer hidden contents. In such cases, sensing is part of reasoning rather than a passive input stream. A simulator that omits these channels can make the benchmark easier by removing the need to infer latent physical properties.

Persistent semantic states and transition rules. The second major requirement is semantic persistence. Many household activities depend on states such as `open`, `filled`, `dirty`, `cooked`, `broken`, or `attached`. These states must persist across time and remain consistent with physical interaction. BEHAVIOR-1K and OmniGibson [216] provide an important foundation in this direction. The benchmark defines 1,000 everyday activities with rich physical and semantic object properties, while OmniGibson supports rigid bodies, deformables, liquids, and transition rules for processes that cannot be represented by pose alone. RoboCasa365 [275] scales the task side of this idea to 365 everyday kitchen tasks across 2,500 environments, with large human and synthetic demonstration collections for multi-task, foundation-model, and lifelong-learning evaluation. The key point is that semantic richness is not simply extra annotation. It determines which causal consequences can be represented and therefore which long-horizon plans can be tested.

Relational state is equally important. An agent may need to track that one object is inside another, that a tool is attached to an end effector, that a container supports an object, or that a previously observed obstacle is now out of view. These relations change after actions and may remain task-relevant long after the

Table 12 Capability comparison of representative simulation platforms.

System	Physical Modeling		World & Task Modeling		Scalability		Validation
	Rigid / Articulated	Deformable / Fluid	Semantic States	Long-Horizon Tasks	Parallel Simulation	Cross-Simulator	Sim-to-Real Validation
OmniGibson [216]	✓	✓	✓	✓	●	✗	●
ManiSkill3 [343]	✓	✗	✗	●	✓	✗	●
MuJoCo [421]	✓	✗	✗	✗	✓	✗	✓
Isaac Lab [269]	✓	●	✗	●	✓	✗	●
RoboVerse [122]	✓	●	●	●	●	✓	●
RoboCasa365 [275]	✓	✗	●	✓	●	✗	✗
MolmoSpaces [197]	✓	●	●	✓	●	✓	✓

✓ Explicit support ● Partial / backend-dependent ✗ Not a primary feature

corresponding objects disappear from the current image. Persistent embodied world models make the same issue explicit from a learned-model perspective. Zhou et al. [450] maintain a persistent 3D memory while predicting future RGB-D observations, reducing contradictions when previously observed regions become unobserved. For AEI evaluation, the external simulator can provide privileged ground truth for these persistent relations, while the agent must reconstruct them from partial observations. This separation is useful because it lets a benchmark test memory and belief maintenance without conflating them with errors in the evaluator.

Scalability, interoperability, and simulation ecosystems. No single simulator is likely to dominate every AEI setting. High-fidelity contact, large-scale rendering, semantic household state, deformable physics, and real-time multi-agent execution impose different computational and engineering requirements. This motivates a shift from monolithic simulators toward composable simulation ecosystems. RoboVerse [122] introduces MetaSim, an abstraction layer that standardizes core operations across simulation backends and supports data generation, benchmarking, and policy learning in a unified platform. MolmoSpaces [197] takes a related ecosystem view at larger scene scale: it provides more than 230,000 indoor environments and 130,000 annotated assets, supports multiple simulators including MuJoCo, Isaac, and ManiSkill, and covers static manipulation, mobile manipulation, navigation, and multi-room long-horizon tasks. The Interactive World Simulator [368] explores another route in which learned interactive world models act as physically consistent surrogates for training and policy evaluation, including rigid and deformable interactions. These systems suggest that AEI infrastructure will increasingly separate assets, physics backends, world-state interfaces, and evaluation logic so that each can evolve independently.

Table 12 compares representative simulation platforms along four dimensions that are particularly relevant to AEI. *Physical modeling* considers whether the simulator supports rigid and articulated objects as well as more complex deformable objects and fluids. *World and task modeling* examines whether it can maintain persistent semantic states and support multi-stage, long-horizon interactions. *Scalability* captures both the ability to run many simulation instances in parallel and the interoperability across different simulator backends. Finally, *validation* asks whether performance measured in simulation has been systematically compared with real-world robot behavior.

These dimensions reveal that the central trade-off is not simply between realism and computing speed. Platforms optimized for high-throughput rigid-body simulation may provide efficient parallel training but limited support for semantic state changes or complex physical interactions. Conversely, task-rich household environments may support persistent states and long-horizon activities while offering less flexibility in physical modeling or large-scale parallelization. No single simulator is therefore optimal for all AEI settings; the appropriate platform depends on the capability being evaluated and the type of evidence required to support that evaluation.

10.2 AEI Benchmark Spectrum: The Cognitive Capability Ladder

Traditional robot benchmarks are often organized by mechanical task type: navigation, grasping, pushing, rearrangement, or locomotion. This taxonomy is useful for controller design, but it is not sufficient for AEI. Two tasks with the same motor primitive can require very different levels of intelligence. Picking up a



Figure 15 The AEI cognitive capability ladder. Each tier is characterized by the *reason* a task is hard rather than by the motor primitive it uses, so that a benchmark result can be attributed to a specific capability. Moving from C1 to C5 increases the information the agent must acquire, the number of intermediate states it must manage, and the extent to which success depends on adaptation rather than direct execution. The bottom row records what a protocol fails to detect when it scores only endpoint success. These are benchmark regimes, not a second autonomy taxonomy; they are distinct from the L0–L5 levels of Table 1.

visible mug from an uncluttered table mainly tests perception and control. Picking a suitable mug when the instruction is ambiguous, the target is occluded, and the agent must first move an obstacle or ask a question tests active perception, uncertainty management, and language grounding. AEI benchmarks should therefore expose the *reason for difficulty*, not only the surface form of the action.

We organize this space as a progressive cognitive capability ladder. The five tiers below are benchmark regimes rather than a second autonomy taxonomy, they should not be confused with the autonomy levels defined earlier in this survey. The tiers increase the amount of information the agent must acquire, the number of intermediate states it must manage, and the extent to which success depends on adaptation rather than direct execution. Recent benchmarks already cover parts of this ladder. EmbodiedBench [401] evaluates capabilities such as commonsense reasoning, spatial awareness, visual perception, instruction understanding, and long-term planning across 1,128 tasks. VLABench [434] similarly combines implicit human intentions, world knowledge, physical laws, and long-horizon language-conditioned manipulation. These benchmarks mark a shift from action-centric evaluation toward capability-centric evaluation. Figure 15 summarizes the five tiers together with the evaluation blind spot associated with each.

C1: Goal disambiguation and active physical grounding. The first capability is recognizing when the current observation is insufficient for making a reliable decision. For example, given the instruction "bring me that cup," an agent should not immediately choose one if several similar cups are visible or if a relevant cup is partially occluded. Instead, it may change viewpoint, inspect the hidden region, move an occluder, or ask the user for clarification; likewise, if the task requires a filled cup but the liquid level is not visible, the agent may need to inspect or manipulate the cup before acting. A benchmark at this level should therefore include ambiguous references, occluded targets, hidden object states, and other situations in which multiple interpretations remain plausible, so that success depends on actively reducing uncertainty rather than confidently guessing. Help-seeking methods such as uncertainty-aligned robot planning [306] already show how ambiguity can trigger interaction with a human rather than an arbitrary choice. VLABench [434] stresses implicit intentions

and physical knowledge, while ENACT [363] evaluates whether models understand how egocentric actions change future states through forward and inverse world-modeling tasks derived from long-horizon BEHAVIOR interactions. Together, these works provide several key ingredients for active grounding: partial observability, action-conditioned state change, and the need to acquire information before committing to an action.

C2: Tool discovery, affordance reasoning, and physical reachability creation. The second capability appears when the agent cannot complete the task through its default embodiment alone. It must reason from the desired physical effect to the required object property, then select or create a tool that extends its affordances. PAC Bench [130] directly tests whether foundation models understand object properties, affordances, and physical constraints that determine whether a manipulation policy is executable. This is a necessary precursor to tool use: a robot cannot choose a useful sponge, hook, or support if it does not understand absorbency, reachability, stability, or graspability.

Tool use becomes more agentic when a suitable tool is not directly available and the agent must identify, adapt, or even create one for the task. RobotSmith [235] couples VLM-based proposals with physics simulation to generate robot-specific tool geometries and usage plans for tasks involving rigid, deformable, and fluid objects. CaP-X [116] exposes a related issue at the software interface. It evaluates coding agents that must synthesize executable manipulation programs from perception and control primitives, and shows that performance depends strongly on how much high-level scaffolding is supplied by the interface. This observation is important for benchmark design. A benchmark should distinguish agent capability from capability hidden in hand-designed primitives. If the benchmark provides a `clean-table()` skill, it does not test whether the agent can discover the sequence of wiping, regrasping, and tool-selection actions needed to clean a table.

C3: Online disturbance handling, reflection, and recovery. A defining property of closed-loop agency is the ability to continue after the world deviates from the plan. Evaluation should therefore inject disturbances during execution rather than only randomize initial states. Relevant events include grasp failure, slip, object displacement, kinematic blockage, unexpected contact, stale perception, and task-state mismatch. The agent must first recognize that execution has failed, then identify the failure stage, select a repair, and decide whether to retry locally or backtrack to an earlier valid state.

VLA-FixBench and FaultEval [392] provide a concrete recent example. They organize manipulation faults into perception, planning, and control failures, annotate task stages and repair strategies, and evaluate VLM-based diagnosis and recovery through static, dynamic, and real-world settings. The associated recovery mechanism localizes the deviation and rolls execution back to enable targeted repair. RoboCerebra [135] evaluates a broader System-2 role in long-horizon manipulation, including planning, reflection, and memory above a low-level controller. Together, these works support a shift from robustness as "succeed despite randomization" to robustness as an observable closed-loop process: detect, explain, revise, and recover. Repeating the same failed action should be scored differently from a successful recovery even if both episodes eventually terminate.

C4: Open-world, long-horizon ecological tasks. In long-horizon tasks, a failure may be caused by an action taken many steps earlier rather than by the agent's most recent action. The agent must maintain task state over time, preserve facts that are temporarily unobserved, respect dependencies among subtasks, and verify that earlier actions remain valid after later changes. BEHAVIOR-1K [216] established a large set of human-centered everyday activities whose execution depends on complex manipulation and long task horizons. RoboCasa365 [275] extends this direction with 365 tasks and 2,500 diverse kitchen environments, enabling controlled studies of multi-task learning, foundation models, environment variation, and lifelong learning.

A long task does not necessarily test whether the agent can remember and use information from earlier steps. RoboMME [94] provides a standardized benchmark for history-dependent manipulation and separates temporal, spatial, object, and procedural memory across 16 tasks. This taxonomy reveals why simply concatenating more atomic actions is not enough to create a strong long-horizon benchmark. A meaningful AEI task should contain delayed dependencies, temporarily hidden state, repeated operations, exceptions, and decisions whose correctness depends on earlier events. Such structure tests whether the agent maintains a usable internal state rather than merely executing a long sequence.

C5: Physical social and multi-agent collaboration. The highest tier in the planning framework concerns embodied collaboration. Here, task success depends on another agent whose actions change the shared

physical state. Human–robot collaboration adds behavioral uncertainty, while multi-robot collaboration adds coupled workspaces, timing constraints, and contact coordination. Moving Out [185] targets physically grounded human–AI collaboration in continuous environments where object mass and geometric constraints affect whether partners can move heavy objects or coordinate around corners. Recent multi-robot work also combines planning, manipulation, and verification agents to close the loop over real multi-robot execution [137]. These studies show the direction of travel, but benchmark coverage at this tier remains less mature than for single-agent manipulation. In particular, standardized evaluation of nonverbal force coordination, shared uncertainty, responsibility allocation, and recovery from partner errors remains limited. This is an important boundary of current AEI evaluation rather than a solved benchmark category.

The tiers are not mutually exclusive. A realistic household mission may combine active grounding, tool use, recovery, long-term memory, and collaboration in one episode. Their value is diagnostic: they make it possible to ask which capability caused success or failure.

10.3 Decoupled Evaluation Metrology & Verifiable Trustworthiness

A single task success rate cannot fully describe how well an AEI agent performs. An agent may eventually complete the task, but only after making poor decisions, causing repeated collisions, or retrying many times. Another agent may correctly detect a failure and revise its plan, but still miss the final goal because of a low-level control error. Treating both cases only as success or failure hides these important differences. Evaluation should therefore measure not only whether the task is completed, but also how the agent reasons, recovers from mistakes, and executes actions during the process.

From endpoint success to process-aware measurement. Let τ denote an interaction trajectory, and let $G(\tau) \in \{0, 1\}$ indicate whether the agent reaches the task goal. Rather than using $G(\tau)$ as the only evaluation result, AEI evaluation should treat it as one part of a broader metric vector:

$$\mathbf{M}(\tau) = [G(\tau), M_{\text{cog}}(\tau), M_{\text{rec}}(\tau), M_{\text{phy}}(\tau), M_{\text{safe}}(\tau)] \,. \tag{9}$$

Here, M_{cog} measures whether the agent makes reasonable decisions during the task, M_{rec} measures how well it detects and recovers from failures, M_{phy} measures the quality of physical execution, and M_{safe} records violations of hard safety constraints. This separation is important because completing a task does not necessarily mean that it was completed well. SafeLab [48], for example, shows that a robot can finish a laboratory task while still causing spills, applying excessive force, or making other irreversible mistakes. OopsieVerse [52] makes a similar distinction in household tasks by explicitly modeling damage caused by contact forces, temperature changes, and liquid interactions. AEI evaluation should therefore report not only whether the goal was reached, but also how the agent reasoned, recovered, and acted during the process.

Cognitive rationality. The first layer measures whether high-level decisions are physically and logically executable. Useful quantities include plan feasibility, impossible-action or hallucination rate, subgoal ordering errors, dead-end frequency, and unnecessary backtracking. These metrics should be computed against explicit environment state whenever possible. For example, attempting to grasp an object outside the reachable workspace is a planning error even if a low-level controller later rejects the command. CaP-X [116] is instructive because it varies the abstraction level of the action interface and shows that apparent coding-agent competence can depend strongly on human-provided primitives. Evaluation should therefore report the action interface and tool budget together with planning metrics; otherwise gains from better scaffolding can be misattributed to better reasoning.

Reflexivity and recovery. The second layer evaluates whether the agent closes the loop after a deviation. Metrics include fault-detection recall, diagnosis accuracy, recovery success, recovery latency, backtracking depth, repeated-error rate, additional actions, and extra model calls. VLA-FixBench [392] provides a structured fault taxonomy across perception, planning, and control, making it possible to measure where diagnosis fails rather than only whether recovery eventually succeeds. RoboCerebra [135] similarly separates planning, reflection, and memory in long-horizon evaluation. Cost should be reported jointly with recovery rate. An agent that calls a large VLM after every action or repeatedly asks for human intervention can obtain high robustness while providing little autonomous value.

Table 13 Decoupled metric matrix for AEI evaluation. Metrics should be selected according to the benchmark claim rather than reported mechanically for every task.

Dimension	Example metrics	What the metrics diagnose
Cognitive rationality	Plan feasibility; impossible-action rate; subgoal-order errors; dead-end frequency; backtracking overhead	Whether high-level reasoning respects geometry, task preconditions, physical constraints, and the provided action interface.
Reflexivity and recovery	Fault-detection recall; diagnosis accuracy; self-recovery success; recovery latency; repeated-error rate; extra actions/model calls	Whether the agent recognizes deviations, identifies their cause, and repairs them without excessive retries or human dependence.
Physical compliance	Jerk; contact-force margin; trajectory length; action/energy cost; collision severity; object damage	Whether execution is mechanically controlled and efficient rather than merely endpoint-correct.
Hard safety boundaries	Forbidden-region violations; force/speed-limit violations; human-proximity events; irreversible hazard count	Whether the trajectory remains inside declared safety and authority constraints throughout execution.

Physical compliance and hard safety boundaries. The third and fourth layers concern the executed trajectory. Physical compliance captures graded properties such as contact force, jerk, trajectory length, energy or action cost, object displacement, and damage severity. Hard safety boundaries capture constraints that should not be crossed: forbidden workspace entry, excessive force or speed, unsafe human proximity, and irreversible hazardous states. These two layers should remain distinct. A smooth but unauthorized trajectory can violate a hard rule, while a safe trajectory can still be wasteful or mechanically poor. Formal constraints such as control barrier functions [36] provide one mechanism for enforcing state-dependent safe sets, but benchmark reporting should still expose intervention frequency, minimum safety margin, and residual risk rather than only whether a safety filter was present.

Multimodal autonomous judging. Open-ended tasks create another difficulty: many success conditions cannot be reduced to Euclidean distance. "Tidy the room," "assemble the object correctly," or "arrange the table for a user" may require semantic judgment. VLM-based judges are therefore attractive, but they should not replace objective state when objective state is available. We recommend a hierarchy of evidence:

$$\text{State} \succ \text{Sensors} \succ \text{Learned Judge} \succ \text{Human}, \tag{10}$$

where $\succ$ indicates preference for the more direct and reproducible source when it can answer the same question. Containment, contact, joint state, force, or temperature should be checked from simulator or sensor state. Learned judges are more appropriate for semantic quality that is not fully captured by such predicates.

RobotArena ∞ [165] illustrates how learned and human evaluation can be combined at scale. It converts real robot demonstrations into simulated counterparts, scores VLA executions with a VLM-guided progress measure, and also collects crowdsourced pairwise preferences. RoboArena [45] provides a complementary real-world protocol: distributed evaluators perform double-blind pairwise comparisons, and the aggregated preferences produce rankings across policies without requiring identical tasks at every site. These approaches are useful when task quality is hard to encode, but learned judges should still be calibrated against human labels and checked for systematic bias. A VLM explanation is evidence about the judge's interpretation, not direct evidence that the physical trajectory was correct.

Anti-gaming and physically valid success. A benchmark is vulnerable when the easiest way to increase score is not the behavior the benchmark intends to measure. In simulation, policies can exploit collision tolerances, interpenetration, unrealistic friction, attachment shortcuts, reset logic, or privileged state. A robust protocol should therefore distinguish task completion from physically valid completion. One simple formulation is

$$S_{\text{valid}}(\tau) = G(\tau)\,\mathbf{1}[C_{\text{physics}}(\tau) \wedge C_{\text{safety}}(\tau)]\,, \tag{11}$$

where G is goal satisfaction and the two constraints reject physically invalid or unsafe trajectories. OopsieVerse [52] operationalizes part of this idea through simulator-agnostic damage signals, while SafeLab [48] reports both ordinary task success and safe success. The broader lesson is that benchmark designers should publish the success predicates, privileged signals used by the evaluator, and known simulator failure modes. Hidden evaluation logic makes it difficult to distinguish improved intelligence from improved exploitation.

Statistical rigor under expensive physical evaluation. Real-robot experiments are usually too expensive for the sample sizes common in computer vision or language modeling. This makes uncertainty reporting part of the evaluation protocol, not a cosmetic addition. Results should include repeated trials, task-stratified performance, and uncertainty estimates. When policies are compared on the same tasks, paired evaluation can reduce variance. Beyond Binary Success [330] develops a sequential, anytime-valid comparison framework that supports partial-credit and continuous metrics and can stop once sufficient statistical evidence has accumulated. AutoEval [451] attacks a different bottleneck by automating real-world success detection and scene reset so that evaluation can run with little human involvement. These directions matter for AEI because richer metrics are useful only if they can be estimated reliably within realistic hardware budgets.

Sim-to-real validity and intelligence rank alignment. Simulation is most useful as an evaluator when conclusions made in simulation predict conclusions in reality. SIMPLER [229] directly studies this question through more than 1,500 paired simulation–real evaluations across two robot embodiments and eight task families, showing that simulated evaluation can recover meaningful trends in real policy performance when visual and control gaps are addressed. PolaRiS [162] extends the real-to-sim approach to generalist policy evaluation using neural scene reconstruction. Across 600 real-world rollouts and more than 93,000 simulation rollouts, it reports stronger correlation with real-world policy ranking than prior simulated benchmarks. RoboLab [404] uses high-fidelity simulation and controlled perturbations to analyze which external factors drive policy behavior and failure. MolmoSpaces [197] reports strong sim-to-real ranking correlation on its benchmark while exposing sensitivities to prompt wording, initial joint state, and camera occlusion. These studies move simulator validation from visual similarity toward behavioral predictiveness.

For AEI, rank agreement should be treated as a minimum requirement rather than the full definition of validity. We propose three complementary checks. *Rank consistency* asks whether policy ordering is preserved, measured for example by Spearman's ρ or Kendall's τ. *Score calibration* asks whether the magnitude of simulated performance is informative about real performance, not only its ordering. *Failure-mode concordance* asks whether a policy fails for similar reasons in simulation and reality. The last criterion is especially important for agentic systems. A simulator that preserves ranking but turns perception failures into contact failures may be adequate for leaderboard construction but weak for diagnosing planning, recovery, or safety.

Taken together, these principles define a stricter standard for AEI evaluation. Simulators should expose the physical and semantic state changes that make agentic behavior necessary. Benchmarks should challenge identifiable capabilities such as active grounding, affordance reasoning, recovery, memory, and physical collaboration. Evaluation protocols should then provide process-level and physically grounded evidence, with uncertainty estimates and real-world validation. Progress toward general embodied autonomy depends not only on building stronger agents, but also on ensuring that the environments, tasks, and measurements used to evaluate them test the capabilities we claim they test.

11 Open Challenges and Future Horizons

11.1 Perception and Grounding

Perception--decision interfaces. How the belief is passed to planners and policies determines what they can check. Five families of interfaces are in use. **Language descriptions** can be read by any language model, but they lose geometry, timing, and uncertainty [156, 425]. **Structured scene state** lets a planner check preconditions, but it cannot represent anything its predefined schema does not include [128, 181, 300]. **Perception as callable tools** makes perception task-conditioned by design. The program decides what to perceive, and every output has a defined type and can be checked [132, 234, 338]. This interface is limited by the modules available, and the apparent skill of coding agents in manipulation depends strongly on how much support the interface provides [116]. **Learned tokens** pass perception to language models and VLA policies as

internal embeddings [60, 64, 103, 196], with variants that add 3D or multisensory input [179, 217, 296, 443]. They are compact and trained together with action, but the belief becomes implicit. Pretrained visual representations aim to supply such tokens for any task [274], but no single representation works best across embodied tasks [259]. Embodied chain-of-thought partly exposes the belief again by having the policy state subtasks and object boxes before it acts [423]. **Unified embodied reasoning models** produce explicit outputs on request. Gemini Robotics-ER predicts points, boxes, trajectories, and correspondences between views, and introduces the ERQA benchmark [346]. Its successor coordinates a VLA policy and checks whether each subtask succeeded [345]. Open models output affordances, spatial references, trajectory traces, and progress estimates [170, 210, 342, 344, 395, 419], or reason about physical common sense in language [277]. Their outputs can be checked, but they are still not calibrated.

The families complement each other. An agent may call tools for precise geometry, keep a scene graph for planning, query an embodied reasoning model for open-ended judgments, and let a VLA policy run a short control loop. Explicit outputs are required wherever the agent must check a precondition, ask a question, or verify an outcome.

Open challenges. The belief view also shows which problems remain open.

- **Calibrated, task-relevant uncertainty.** Foundation models rarely provide a confidence that predicts their own errors. The central open problem of agentic perception is to produce calibrated sets of candidates, with an option to abstain, for the variables that matter to the decision at hand.
- **Metric and frame-consistent spatial reasoning.** Geometric models now recover 3D structure well, but language models still reason poorly about it [397, 402, 439], especially in robot-centric and multi-camera settings [114].
- **Physical understanding beyond vision.** Tactile and force data remain scarce and specific to each sensor, and physical benchmarks show large gaps for current models [56, 89].
- **Persistent belief in changing environments.** A belief must record when its contents may be out of date, and it must keep track of object identity across days, sessions, and robots.
- **When and how to gather information.** Choosing between looking, touching, and asking requires estimating both the value of the information and its cost, including physical risk.
- **Inspectable end-to-end policies.** VLA policies should be able to report, on request, the referents, poses, and states they rely on, together with their confidence.
- **Evaluation that matches the belief view.** Most benchmarks test one component in isolation (Table 2), and their validity must be checked first. A text-only model that exploits patterns in a benchmark's questions can score well on spatial benchmarks without seeing the images, so results should be compared with such blind baselines and reported on debiased splits [69]. The belief view adds four requirements. First, errors should be judged relative to the task, since an error of two centimeters is harmless for navigation and fatal for grasping. Second, confidence should predict correctness. Conformal methods make this practical [304, 306], but abstention benchmarks show that current models are far from it [381]. Third, tests should check whether the belief is still correct after unobserved changes, not only whether past observations are recalled [94, 258]. Fourth, the cost of information, measured in viewpoints, contacts, and questions, should be reported alongside success. Integrated benchmarks such as ERQA, EmbodiedBench, and ENACT test several components together [346, 363, 401]. Sections 10.2 and 10.3 place perception within capability-oriented benchmarks and process-aware metrics.

11.2 Reasoning and Planning

The preceding sections suggest that reliable long-horizon planning depends on the connections between decomposition, search, representation, and execution. A plausible decomposition can produce infeasible subgoals; a search procedure can prefer an inaccurate predicted future; and an executable representation can omit conditions needed by the underlying policy. We identify four directions for strengthening these connections, with particular emphasis on branch selection and the coupling between high-level reasoning and motor skills.

Reliable branch selection under limited computation. The central search problem is deciding which alternatives merit further evaluation and which evidence justifies committing to an action. Tree-Planner, VLP, and ParallelWorld illustrate different ways to organize candidate branches or evaluate prospective outcomes [81, 104, 144]. Additional search is useful only when its evaluations discriminate between outcomes that matter for the task. In embodied environments, visually plausible predictions may omit a failed contact, an unreachable configuration, or uncertainty about an occluded object. Deeper rollouts can compound these errors, while aggressive pruning can remove a necessary preparation or observation action whose value emerges only later. An important direction is uncertainty-aware allocation of search depth and width: expand branches when additional evidence could change the decision, preserve alternatives when estimates are unreliable, and stop when further computation is unlikely to improve action selection. Verifiers should be evaluated against observed outcomes and task constraints, including their ability to recognize unsupported predictions. Such evaluation would help distinguish confidence in a generated future from evidence that the corresponding action is executable.

Capability-aware decomposition and planner--policy coupling. High-level planning must account for the specific motor policies available to the robot. SayCan grounds skill selection in affordance estimates, Text2Motion considers geometric feasibility across skill sequences, and RoboHarness explicitly addresses heterogeneous policy handoffs [28, 151, 236]. These perspectives motivate a tighter connection between subgoal generation and execution capability. A planner should be able to query whether a policy can attempt a proposed subgoal, which parameters or starting states it supports, and what distribution of terminal states it is likely to produce. The latter is particularly important: successful completion of one skill need not establish the conditions required by the next. Future systems could use policy-conditioned feasibility estimates and outcome models to jointly choose subgoal granularity, skill identity, and handoff states. Execution feedback should also distinguish failures of grounding, reachability, contact, or goal interpretation, so that the planner can revise the relevant part of the plan. This would make low-level capability an input to planning, rather than information revealed only by unsuccessful execution.

Verifiable interfaces across planning representations. Language plans, generated programs, and callable skills expose different aspects of a task. Translating between them can lose object identity, ordering constraints, success conditions, or assumptions about the scene. Programmatic and constraint-based approaches provide useful mechanisms for making some of these requirements explicit [82, 153, 234], but no single representation guarantees physical validity. A promising direction is to preserve a compact task contract across representations: grounded targets, preconditions, expected effects, termination criteria, and recovery options. These contracts would allow a planner to compare what was intended, what was requested from the executor, and what was observed. They should also accommodate uncertain or learned conditions, rather than treating every predicate as perfectly known. The research challenge is to obtain interfaces that remain inspectable and checkable without requiring an exhaustive hand-written model of every skill and environment.

Evaluation that separates planning quality from execution resources. End-to-end success alone provides limited insight into why a planning system works. Improvements can arise from better decomposition or branch selection, stronger motor policies, additional observations, or larger inference and retry budgets. Future evaluation should compare planners under matched execution capabilities and explicitly account for computation and interaction costs. Diagnostic settings, such as those explored by Embodied Agent Interface [223], can be complemented with interventions that change skill availability, invalidate an intermediate condition, or reveal new information during execution. Useful measures include subgoal feasibility, branch-selection quality when reference outcomes are available, handoff success, recovery success, and decision latency. Tests should also distinguish access to restorable simulator states from planning based solely on observations and learned predictions, since these provide different information and computational advantages.

Together, these directions point toward planners that search selectively, generate subgoals with explicit knowledge of execution capabilities, and retain verifiable task semantics throughout the planning–execution loop. Progress should be assessed by whether these connections improve reliable physical task completion under realistic sensing, computation, and interaction budgets.

11.3 Action Execution

Recent advances in short-horizon action policies have enabled increasingly generalizable and expressive robot control through multimodal conditioning, flexible action representations, and generative action models. Together with closed-loop and asynchronous execution, these developments have improved the ability of learned policies to translate high-level task objectives into continuous robot actions. Nevertheless, several challenges remain for reliable deployment in complex real-world environments.

Generalizable Dexterous Manipulation. Despite recent advances in generalist robot policies, many existing VLA models primarily focus on manipulation with two-finger grippers [66, 196, 279]. Extending these policies to high-DoF dexterous hands remains challenging due to complex finger coordination, high-dimensional action spaces, and uncertain contact dynamics. Recent efforts, such as DexVLA [376] and DexUMI [389], demonstrate promising progress in real-world dexterous manipulation, yet achieving robust generalization across diverse objects, tasks, and hand embodiments remains an open challenge. Future research should focus on scalable dexterous manipulation data, efficient hand-action representations, and contact-aware closed-loop policies that integrate visual and tactile feedback. These advances could enable generalist policies to move beyond simple pick-and-place tasks toward more sophisticated dexterous skills, including in-hand manipulation and fine-grained object interactions.

Cross-Embodiment Generalization. Cross-embodiment generalization remains challenging due to differences in robot morphologies, kinematic structures, and action spaces, highlighting the importance of unified robot representations. Recent approaches explore shared action spaces [416], robot-object distance representations [372], relative spatial transformations [111], and contact-based representations [383] to facilitate knowledge transfer across heterogeneous embodiments. Despite these advances, existing representations still face challenges in capturing shared manipulation behaviors while preserving embodiment-specific geometric, kinematic, and contact constraints. Future research should prioritize more expressive and interaction-centric representations that jointly encode robot morphology, kinematics, and physical interactions with the environment, enabling effective zero-shot transfer to unseen embodiments with minimal robot-specific adaptation.

Efficient VLA Deployment and Real-Time Execution. Deploying large-scale VLA models on physical robots remains challenging due to their substantial computational demands, memory requirements, and inference latency, particularly on resource-constrained robotic platforms. Recent approaches improve execution efficiency through compact model architectures, efficient action decoding, and asynchronous inference [61, 194, 287, 325]. However, achieving low-latency and stable inference while maintaining strong policy performance remains an open challenge. Future research should emphasize system-level infrastructure optimization, including model compression and quantization, efficient memory management, feature caching, and hardware-aware inference scheduling. Furthermore, tightly integrating perception, policy inference, and robot control through optimized execution pipelines could reduce end-to-end latency and enable more reliable real-time deployment of generalist robot policies.

11.4 Closed-loop Evolution

Evaluation must follow the update target. For control correction, report detection recall, false alarms, latency, recovery success among detected failures, and end-to-end recovery over all failures. For reflection, test whether the diagnosis changes a checkable intervention and whether that intervention works under a matched retry budget. For memory and skills, clear transient context and measure reuse on new objects, scenes, and embodiments. For model and system updates, report held-out performance for every version, retention of earlier tasks, transfer, rejected candidates, and cumulative interaction, demonstration, compute, and human-reset costs. Longitudinal curves over $S_0, S_1, \ldots, S_K$ are more informative than the best final checkpoint.

Current protocols expose different pieces of the loop: SWE-bench assesses issue resolution, RoboFail tests explanation and recovery, Voyager tests continuing skill acquisition, and RoboCat evaluates manipulation transfer and adaptation [63, 177, 247, 355]. None alone establishes open-ended physical self-improvement. A defensible study separates improvement-generation, candidate-validation, and held-out test tasks; matches observation and interaction budgets; includes fixed-retry, no-memory, no-reflection, no-external-verifier, and

equal-budget offline-retraining baselines; and counts safety violations, near misses, human intervention, and hardware time over all trials. Real-robot studies should also randomize trial order, repeat sessions, and record initial conditions so that learning can be distinguished from operator practice and hardware drift.

The principal open problems follow from this pipeline. *Cross-layer diagnosis* must distinguish failures with similar symptoms but different causes through event representations and selective interventions. *Memory and skill applicability* must survive changes in embodiment, environment, and time with explicit preconditions, provenance, uncertainty, and conflict resolution. *Independent evaluation* must remain reliable as learned evaluators, simulators, and generated rewards become optimization targets. *Resource-aware persistence* must prevent new skills from erasing old ones while accounting for human labor and failed physical trials. Finally, *safe system evolution* must connect every proposed change to authority limits, runtime safeguards, replay, regression testing, and rollback.

Closed-loop evolution is therefore a discipline for selecting the right change after an embodied failure. Some failures require another observation, some a repaired plan, some a reusable skill, and others a new policy or training procedure. Progress should be measured by whether execution evidence produces a targeted, transferable, independently validated update, rather than by whether an agent generates a convincing reflection or obtains a higher score after an unspecified amount of additional interaction.

11.5 Computing Infrastructure

The chapter's claim in one sentence: **the infrastructure question of the end-to-end era was a performance problem---execute one model efficiently; in the agentic era it becomes a systems problem---manage a fleet of models, protocols, memories, and robots.** We close with an explicit gap map (Fig. **??**): a five-row inventory of the built and the unbuilt, including a cross-cutting row for safety and failure modes—a topic conspicuous by its absence, given that fleet scheduling wired to the physical world through network-callable tools converts every software fault into a potential physical incident (Chapter 8 develops this theme).

Each open cell is an immediately tractable direction, and each is difficult for an identifiable reason. *Learned split policies*: the state space (energy, bandwidth, load) is easy to simulate but hard to transfer to production networks; a first paper resembles sim-to-real of a splitting controller trained against trace-driven network models. *Millisecond guarantees for robot MCP ecosystems*: tool calls inherit no real-time semantics; a first paper resembles an MCP gateway that arbitrates tool invocations against control deadlines. *Graceful degradation under network loss*: the onboard stack must absorb a role it was not provisioned for; a first paper resembles a provably bounded fallback policy with cloud re-synchronization. *Robotics-specific silicon*: it awaits workload stabilization—what the agentic loop actually computes per step—which is itself an open measurement question. *Safety and failure modes*: injection-resistant tool boundaries and physical permission models; a first paper resembles a formal account of what an agent must never be permitted to invoke.

Finally, three interfaces to the remaining chapters. To the action-execution chapter: the latency budget provided by infrastructure dictates policy selection. To the data-flywheel chapter: the device–cloud link of this chapter is the flywheel's return path, with data factories [19] as its current form. To the evaluation chapter: protocols must model deployment explicitly; otherwise a model topping cloud benchmarks may fail on the device.

11.6 Safety and Trustworthiness

Agentic embodied safety has moved beyond collision avoidance and instruction refusal. Current systems can recognize some contextual and relational hazards, convert selected natural-language requirements into planning constraints, filter physically infeasible actions, ask for help under calibrated uncertainty, and carry safety-relevant experience across episodes [249, 306, 326, 406, 429]. These results support a measured conclusion: agentic mechanisms add useful safety capabilities because they can reason over prerequisites, intermediate states, and history. They do not yet provide a general end-to-end guarantee. Existing evidence is usually tied to a particular hazard ontology, embodiment, dynamics model, memory representation, or benchmark, and improvements in one layer do not automatically compose into a safe closed loop.

The most important next step is to connect the layers with explicit contracts and independent evidence. A

robust system should ground risk in observable physical state, decompose long-horizon tasks into verifiable subgoals, enforce authority and dynamics constraints at runtime, record the provenance and applicability of safety memories, and gate every update with replay, red-team testing, regression checks, and rollback. Until such cross-layer, cross-embodiment, long-horizon, and real-world evaluations become routine, the appropriate claim for agentic embodied AI is improved safety capability under stated assumptions, rather than solved safety.

11.7 Agentic Data Ecosystem

Open Challenges. Connecting heterogeneous data into usable experience remains difficult. Human videos, robot trajectories, generated rationales, and diagnostic annotations differ in temporal resolution, embodiment, and annotation quality. Linking a decision to its execution interval and consequences requires aligned timestamps, entity references, and annotation provenance. Errors also propagate through the flywheel: incorrect outcome labels affect policy updates and subsequent task selection, while collection concentrated on successful or familiar tasks leaves rare failures and recovery behaviors underrepresented.

Physical collection further incurs safety constraints, equipment wear, and human reset costs. Safe Exploration [95] and the Safe Learning in Robotics survey [70] examine constraints on learning through physical interaction. Simulation expands collection, but differences in contact dynamics, sensing, and embodiment complicate transfer to real robots. These costs and mismatches constrain how broadly and how often an experience flywheel can operate.

Future Horizons. A useful next step is an event-linked experience record connecting goals, decisions, executed actions, multimodal feedback, diagnoses, interventions, and subsequent outcomes. Shared entity identifiers and versioned annotations would support retrieval across attempts, correction of disputed labels, and reuse in later training rounds. Such records would make the experience schema operational across collection and learning pipelines.

Data coverage should extend toward rare failures and complete recovery episodes. Records should capture anomaly onset, diagnostic probes, safe fallbacks, corrective actions, and independently checked outcomes. VLA-Fix [392] and OopsieVerse [52] provide relevant settings for fault diagnosis and recovery, and damage-aware evaluation, respectively. Retaining unsuccessful recovery attempts would also expose when a diagnosis or proposed correction needs revision.

For **L4 (High Domain Autonomy)**, data should capture long-horizon mission execution within the operational design domain (ODD), autonomous recovery, ODD-boundary detection, and safe halting beyond that boundary. Exception records should link asynchronous tickets and remote responses to subsequent mission outcomes, supporting evaluation of operation with zero on-site physical intervention.

For **L5 (Full General Autonomy)**, the data frontier extends to open-world commonsense reasoning, tool-use transfer, active perception, and lifelong learning. Experience should connect unfamiliar situations to exploratory actions, transferred skills, and verified outcomes. Longitudinal records should track capability retention, adaptation, and intrinsic self-defense under changing conditions, alongside natural interaction through high-level goals and preferences that requires no technical expertise from users.

11.8 Simulators, Benchmarks and Evaluation Protocols

Current evaluation practice is calibrated for conditional autonomy: an episode starts from a scripted reset, ends at a success predicate, and assumes that a human restores the world afterwards. The upper levels of the autonomy ladder invalidate each assumption. L4 asserts that a system may run unattended inside a declared operational domain and halt safely outside it; L5 removes the boundary altogether. Both shift the measurement target from task competence to the conditions under which autonomy can be granted.

Measuring the operating envelope rather than the task. An L4 claim is a claim about an operational domain, not about a task list, yet almost no embodied benchmark publishes the domain over which its results are supposed to hold. The relevant quantity is not in-domain success but boundary behavior: whether the agent recognizes that it has left the conditions under which it is competent, stops, and reports the exception in a

form a remote operator can act on. Adjacent industries address this through explicit domain specification and hazards attributed to performance limitations rather than component faults [219], but embodied benchmarks have not followed. The ingredients exist as components — abstention and uncertainty filtering supply a stopping signal [315, 381], runtime monitors detect deviation under deployment shift [24, 298] — but have not been promoted to deployment-level reporting. A useful direction is to report two error types separately and with asymmetric weight: missed exceptions, where the agent acts outside its competence, and unnecessary escalations, where it defers work a human expected it to absorb. Intervention rate is the natural companion to success rate, but is interpretable only when the detector and its calibration set are specified alongside it [193]. Protocols should further report time to reach a safe state and whether the emitted exception record sufficed for recovery, consistent with arguments that embodied benchmarks should measure deployment reliability rather than accuracy [388].

From episodes to deployments. Resetting the world between trials removes precisely the failures that decide whether unattended operation is viable: irreversible state changes, accumulated clutter, calibration drift, and corruption of the agent's own memory. Long-horizon suites extend task length but remain episodic [94, 216, 275], and the reset itself is increasingly treated as part of the evaluated system rather than free infrastructure [174, 451]. Self-evolving systems (Section 6) add a further difficulty: the system under test changes during the test, making a frozen checkpoint on a frozen suite an ill-posed measurement. Deployment-scale protocols would instead score uninterrupted multi-day operation without external reset, counting cumulative irreversible damage [48, 52] and autonomous recovery rather than per-episode success. For self-modifying agents the central question is regression rather than peak performance: whether competence acquired on new tasks is retained on old ones, whether an upgrade can silently remove a safety behavior, and whether self-assigned rewards continue to track independently measured performance [295, 330].

Benchmarks that cannot be enumerated. A fixed task suite is itself an operational domain, so a static benchmark can certify L4 within its scope but can never evidence the open-world generality that L5 asserts. Finite suites also saturate and leak into training data, and a capable agent is precisely the one able to find the shortcuts that Section 10.3 warns about. This argues for treating evaluation as a generative process rather than a dataset. Recent systems synthesize tasks in situ from the scene the agent is placed in, removing contamination and exposing failures that public leaderboards conceal [136], and agentic pipelines now assemble assets, goals, and verification code end to end [171, 207]. Two ingredients remain underdeveloped: validity of the generator itself, since automatically produced tasks must be physically solvable, semantically meaningful, and discriminative [446]; and adversarial intent, since a generator tuned for diversity does not search for the cases a particular agent gets wrong, unlike inverse-design methods that construct corner cases from a specified failure condition [276]. Results would then be reported as a curve against a measured distance from the training distribution rather than a single number. The collaborative tier is especially exposed here, since partner behavior is an unbounded source of variation that no fixed suite enumerates [137, 185].

Certifying rare failures under physical sampling limits. The binding constraint above L4 is the tail rather than the mean, and exhaustive testing over a high-dimensional space of operating conditions is intractable, so evidence must come from search concentrated on failure boundaries rather than from sampling nominal behavior. Simulation must therefore carry evidential weight, yet sim-to-real validation establishes mainly rank agreement on common tasks [162, 229, 404], which says little about rare events; learned world models sharpen this, since a surrogate fitted to observed data is most accurate exactly where failures are least likely [368, 450]. A more promising posture treats simulator error as part of the inference, yielding evaluation guarantees that remain valid under an imperfect simulator [46]. The verification layer is itself unvalidated, and no single checker covers the space of robot-policy failures [354]. A tiered argument therefore seems necessary: boundary-focused search in simulation, a transfer argument based on failure-mode concordance rather than ranking alone, anytime-valid sequential testing on hardware [330], and an explicit account of residual risk. Until such arguments become standard, evaluation infrastructure remains the gate on autonomy claims: a report that records no observed catastrophic failure, without showing that such failures were actively searched for, is not evidence of full autonomy.

12 Conclusion

This survey has examined agentic embodied intelligence as a system-level transition from conditional action generation toward sustained, feedback-driven physical behavior. The central requirement is not simply to produce a plausible action sequence, but to maintain a task-relevant account of the world, reason about goals and constraints over time, execute through policies and control interfaces, and use observed outcomes to continue, recover, or revise the plan. Perception and grounding, reasoning and planning, action execution, and closed-loop evolution are therefore best understood as coupled functions linked by explicit information and control interfaces. Their organization may be modular or partly internalized by a unified model; in either case, physical evidence must remain capable of changing what the agent believes and does next.

The surveyed approaches provide important parts of this foundation. VLA policies transfer multimodal representations into robot actions, while world-action models introduce predictive structure that can help relate actions to future states; programmatic approaches compose available tools and skills into longer procedures [64, 147, 195, 196, 234]. These capabilities expand the range of instructions and behaviors that robots can attempt, but do not alone establish reliable autonomy. Their performance depends on the surrounding infrastructure that schedules deliberation and control, mediates skill and tool calls, preserves state, and handles communication and exceptions. It also depends on data that connect goals and decisions to physical execution, feedback, and correction, and on evaluation that tests recovery, uncertainty, safety, and transfer rather than final success alone [57, 401].

The autonomy ladder clarifies the remaining gap. Teleoperation, assisted primitives, and bounded single-task policies are established operating modes; language-conditioned foundation policies extend capability but remain conditional on their training and operating conditions. L4 requires sustained mission execution within a declared operational domain, including verification, routine recovery, boundary recognition, and safe escalation. L5 remains an open-world research horizon. The literature reviewed here does not establish sustained unattended L4 operation across a broad, independently validated domain, much less L5 generality. An agentic closed loop is a necessary systems condition for progressing beyond conditional autonomy, but its presence alone cannot guarantee competence: the component models, interfaces, infrastructure, and safety controls must each be validated under the conditions in which the system will operate.

Progress will require calibrated beliefs and active information gathering; plans that account for the actual capabilities and likely outcomes of available policies; low-latency coordination across semantic reasoning and physical control; and memory or model updates accepted only when their benefits and regressions are independently checked. Data ecosystems should preserve failures, interventions, and recovery outcomes alongside successful demonstrations. Simulators and benchmarks should support long-horizon and deployment-level evaluation, including operating-domain boundaries and evidence of safe stopping, while physical tests establish what simulation cannot. Safety must remain a design and evaluation concern throughout perception, planning, execution, adaptation, and deployment.

The path from foundation models to physical agency is therefore a joint challenge in learning, control, infrastructure, data, and assurance. Model scale can broaden what an agent recognizes or attempts; dependable embodied autonomy will depend on whether the complete system can ground its decisions, act within physical limits, detect when reality diverges from expectation, and respond appropriately. Agentic embodied intelligence offers a framework for organizing that work and for measuring progress toward useful, bounded autonomy and, ultimately, more general physical agency.

Authors

Shengjun Zhang, Simin Huang, Chensheng Dai, Cheng Nian, Chong Xia, Xin Fei, Diankun Wu, Weichen Lyu, Min Chen, Mingyu Dong, Zhang Zhang, Yiwen Shen, Yueqi Duan

References

[1] A fog robotics approach to deep robot learning. In *ICRA*, 2019. arXiv:1903.09589.

[2] FogROS: An adaptive framework for automating fog robotics deployment. In *IEEE Int'l Conf. on Automation Science and Engineering (CASE)*, 2021. arXiv:2108.11355.

[3] DeeR-VLA: Dynamic inference of multimodal large language models for efficient robot execution. In *NeurIPS*, 2024. arXiv:2411.02359.

[4] HiRT: Enhancing robotic control with hierarchical robot transformers. In *CoRL*, 2024. arXiv:2410.05273.

[5] RoboOS: A hierarchical embodied framework for cross-embodiment and multi-agent collaboration, 2025. arXiv preprint arXiv:2505.03673.

[6] RoboOS-NeXT: A unified memory-based framework for lifelong, scalable, and robust multi-robot collaboration, 2025. arXiv preprint arXiv:2510.26536.

[7] ROSBag MCP server: Analyzing robot data with LLMs for agentic embodied AI applications, 2025. arXiv preprint arXiv:2511.03497.

[8] Agentic artificial intelligence: Architectures, taxonomies, and evaluation of large language model agents, 2026. arXiv preprint arXiv:2601.12560.

[9] AI-IoT-robotics integration: Survey of frameworks, emerging trends, and the path toward connected robotics. *IEEE Internet of Things Journal*, 2026. arXiv:2606.01015.

[10] Casting everything to online API services? A survey of integrating localized speech recognition models in robotics. In *Int'l Conf. on Social Robotics (ICSR)*, 2026. arXiv:2607.11792.

[11] CLIFT: Turning Gemini Robotics on-device into humanoid specialists via non-invasive closed-loop iterative fine-tuning, 2026. arXiv preprint arXiv:2607.29172.

[12] A comprehensive review of generative physical artificial intelligence. *IEEE Internet of Things Journal*, 2026. arXiv:2609.18111.

[13] ComVLA: Communication-aware split inference for VLA models in 6G-connected robotics. In *IEEE GLOBECOM*, 2026. arXiv:2609.07838.

[14] Diffusion language models for mobile edge agentic AI: Foundations, applications, and challenges, 2026. arXiv preprint arXiv:2609.04778.

[15] DriveMCP: An agentic AI framework for advanced driver assistance system, 2026. arXiv preprint arXiv:2609.17247, submitted to IEEE T-IV.

[16] EcoVLA: Energy-efficient device-edge co-inference for VLA models under real-time constraint. In *Int'l Conf. on Advanced Parallel Processing Technologies (APPT)*, 2026. arXiv:2608.15502.

[17] Efficient vision-language-action management and serving for robot factories, 2026. arXiv preprint arXiv:2609.12075.

[18] FASA: Feedback-aware sampling adaptation for efficient diffusion-based VLA models. In *Int'l Conf. on Networking, Architecture, and Storage (NAS)*, 2026. arXiv:2609.19475.

[19] The robot data factory, 2026. arXiv preprint arXiv:2609.16705.

[20] Towards efficient large language model serving: A survey on system-aware KV cache optimization. In *Findings of ACL*, 2026. arXiv:2607.08057.

[21] VLA-ULAP: Interleaving cloud VLA calls with ultra-lightweight local action prediction at the edge, 2026. arXiv preprint arXiv:2609.18663.

[22] Pieter Abbeel, Adam Coates, and Andrew Y Ng. Autonomous helicopter aerobatics through apprenticeship learning. *The International Journal of Robotics Research*, 29(13):1608–1639, 2010.

[23] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning*, pages 22–31, 2017.

[24] Christopher Agia, Rohan Sinha, Jingyun Yang, Zi ang Cao, Rika Antonova, Marco Pavone, and Jeannette Bohg. Unpacking failure modes of generative policies: Runtime monitoring of consistency and progress. arXiv preprint arXiv:2410.04640, 2024.

[25] Pulkit Agrawal, Ashvin Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to poke by poking: Experiential learning of intuitive physics. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

[26] Ossama Ahmed, Frederik Träuble, Anirudh Goyal, Alexander Neitz, Yoshua Bengio, Bernhard Schölkopf, Manuel Wüthrich, and Stefan Bauer. CausalWorld: A Robotic Manipulation Benchmark for Causal Structure and Transfer Learning, 2020.

[27] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[28] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[29] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[30] Michael Ahn, Debidatta Dwibedi, Chelsea Finn, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Karol Hausman, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, et al. AutoRT: Embodied Foundation Models for Large Scale Orchestration of Robotic Agents. *arXiv preprint arXiv:2401.12963*, 2024.

[31] Anurag Ajay, Seungwook Han, Yilun Du, Shuang Li, Abhi Gupta, Tommi Jaakkola, Josh Tenenbaum, Leslie Kaelbling, Akash Srivastava, and Pulkit Agrawal. Compositional foundation models for hierarchical planning. *Advances in Neural Information Processing Systems*, 36:22304–22325, 2023.

[32] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[33] Yuval Alaluf, Elad Richardson, Sergey Tulyakov, Kfir Aberman, and Daniel Cohen-Or. Myvlm: Personalizing vlms for user-specific queries. arXiv preprint arXiv:2403.14599, 2024.

[34] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. Active vision. *International Journal of Computer Vision*, 1(4):333–356, January 1988.

[35] Matthias Althoff and John M. Dolan. Online verification of automated road vehicles using reachability analysis. *IEEE Transactions on Robotics*, 30(4):903–918, 2014.

[36] Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 18th European Control Conference (ECC)*, pages 3420–3431, 2019.

[37] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pages 3674–3683. IEEE, 2018.

[38] Anthropic. Claude opus 5.5. https://www.anthropic.com/claude-opus-5-5, 2026.

[39] Antonis Antoniades, Albert Örwall, Kexun Zhang, Yuxi Xie, Anirudh Goyal, and William Wang. SWE-Search: Enhancing Software Agents with Monte Carlo Tree Search and Iterative Refinement. *arXiv preprint arXiv:2410.20285*, 2024.

[40] Antonis Antoniades, Albert Örwall, Kexun Zhang, Yuxi Xie, Anirudh Goyal, and William Wang. Swe-search: Enhancing software agents with monte carlo tree search and iterative refinement. In *International Conference on Learning Representations*, volume 2025, pages 64485–64515, 2025.

[41] Abrar Anwar, John Welsh, Joydeep Biswas, Soha Pouya, and Yan Chang. Remembr: Building and reasoning over long-horizon spatio-temporal memory for robot navigation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2838–2845. IEEE, 2025.

[42] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[43] W Ross Ashby and JR Pierce. An introduction to cybernetics. *Physics Today*, 10(7):34–36, 1957.

[44] Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba, Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. arXiv preprint arXiv:2506.09985, 2025.

[45] Pranav Atreya, Karl Pertsch, Tony Lee, Moo Jin Kim, Arhan Jain, Artur Kuramshin, Cyrus Neary, Edward S. Hu, Kanav Arora, Kirsty Ellis, Luca Macesanu, Matthew Leonard, Meedeum Cho, Ozgur Aslan, Shivin Dass, Tony Wang, Xingfang Yuan, Abhishek Gupta, Dinesh Jayaraman, Glen Berseth, Kostas Daniilidis, Roberto Martín-Martín, Youngwoon Lee, Percy Liang, Chelsea Finn, and Sergey Levine. Roboarena: Distributed real-world evaluation of generalist robot policies. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 336–364. PMLR, 2025.

[46] Apurva Badithela, David Snyder, Lihan Zha, Joseph Mikhail, Matthew O'Kelly, Anushri Dixit, and Anirudha Majumdar. Reliable and scalable robot policy evaluation with imperfect simulators. arXiv preprint arXiv:2510.04354, 2025.

[47] Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from human videos as a versatile representation for robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[48] Fengshuo Bai, Yufeng Li, Ruihai Wu, Peishuo Wang, Yuhan Wang, Bernie Hao Zhu, Yuanfei Wang, Tawei Chou, Jing Gao, Runchuan Zhu, Ying Wen, Yaodong Yang, and Yuanpei Chen. Safelab: An interactive high-fidelity benchmark for embodied safety in scientific robotics. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 5324–5347. PMLR, 2026.

[49] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in Time: A Joint Video and Image Encoder for End-to-End Retrieval. *arXiv preprint arXiv:2104.00650*, 2021.

[50] Ruzena Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988.

[51] Ruzena Bajcsy, Yiannis Aloimonos, and John K. Tsotsos. Revisiting active perception. arXiv preprint arXiv:1603.02729, 2016.

[52] Arnav Balaji, Arpit Bahety, Sriniket Ambatipudi, Daniel Lam, Junhong Xu, and Roberto Martín-Martín. Oopsieverse: A safety benchmark with damage-aware simulation for robot manipulation. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026.

[53] Fabien Baradel, Natalia Neverova, Julien Mille, Greg Mori, and Christian Wolf. CoPhy: Counterfactual Learning of Physical Dynamics. In *International Conference on Learning Representations*, 2020.

[54] Lawrence W Barsalou. Perceptual symbol systems. *Behavioral & Brain Sciences*, 22(4):577, 1999.

[55] Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. arXiv preprint arXiv:1612.00222, 2016.

[56] Daniel M. Bear, Elias Wang, Damian Mrowca, Felix J. Binder, Hsiao-Yu Fish Tung, R. T. Pramod, Cameron Holdaway, Sirui Tao, Kevin Smith, Fan-Yun Sun, Li Fei-Fei, Nancy Kanwisher, Joshua B. Tenenbaum, Daniel L. K. Yamins, and Judith E. Fan. Physion: Evaluating physical prediction from vision in humans and machines. arXiv preprint arXiv:2106.08261, 2022.

[57] Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, Hongyan Zhao, Hanyu Liu, Zhizhong Su, Lei Ma, Hang Su, and Jun Zhu. Motus: A Unified Latent Action World Model. *arXiv preprint arXiv:2512.13030*, 2025.

[58] Jianxin Bi, Kevin Yuchen Ma, Ce Hao, Mike Zheng Shou, and Harold Soh. Vla-touch: Enhancing vision-language-action models with dual-level tactile feedback. arXiv preprint arXiv:2507.17294, 2025.

[59] Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 17–40. PMLR, 2025.

[60] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. *pi_0*: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[61] Kevin Black, Manuel Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[62] Jeannette Bohg, Karol Hausman, Bharath Sankaran, Oliver Brock, Danica Kragic, Stefan Schaal, and Gaurav Sukhatme. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics*, 33:1273–1291, 2017.

[63] Konstantinos Bousmalis, Giulia Vezzani, Dushyant Rao, Coline Devin, Alex X. Lee, Maria Bauza, Todor Davchev, Yuxiang Zhou, Agrim Gupta, Akhil Raju, et al. RoboCat: A Self-Improving Generalist Agent for Robotic Manipulation. *Transactions on Machine Learning Research*, 2023.

[64] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. *arXiv preprint arXiv:2307.15818*, 2023.

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

[66] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[67] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. RT-1: Robotics Transformer for Real-World Control at Scale. *arXiv preprint arXiv:2212.06817*, 2022.

[68] Rodney A Brooks. Intelligence without representation. *Artificial intelligence*, 47(1-3):139–159, 1991.

[69] Ellis Brown, Jihan Yang, Shusheng Yang, Rob Fergus, and Saining Xie. Benchmark designers should "train on the test set" to expose exploitable non-visual shortcuts. arXiv preprint arXiv:2511.04655, 2026.

[70] Lukas Brunke, Melissa Greeff, Adam W. Hall, Zhaocong Yuan, Siqi Zhou, Jacopo Panerati, and Angela P. Schoellig. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5:411–444, 2022.

[71] Finn Lukas Busch, Timon Homberger, Jesús Ortega-Peimbert, Quantao Yang, and Olov Andersson. One map to find them all: Real-time open-vocabulary mapping for zero-shot multi-object navigation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14835–14842. IEEE, 2025.

[72] Roberto Calandra, Andrew Owens, Dinesh Jayaraman, Justin Lin, Wenzhen Yuan, Jitendra Malik, Edward H Adelson, and Sergey Levine. More than a feeling: Learning to grasp and regrasp using vision and touch. *IEEE Robotics and Automation Letters*, 3(4):3300–3307, 2018.

[73] Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós. Orb-slam3: An accurate open-source library for visual, visual-inertial and multi-map slam. arXiv preprint arXiv:2007.11898, 2021.

[74] Junyi Cao, Yian Wang, Ziyan Xiong, Chunru Lin, Zhehuan Chen, and Chuang Gan. Dlo-lab: Benchmarking deformable linear object manipulations with differentiable physics. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 11581–11605. PMLR, 2026.

[75] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, Rishi Hazra, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollár, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. Sam 3: Segment anything with concepts. arXiv preprint arXiv:2511.16719, 2026.

[76] Devendra Singh Chaplot, Dhiraj Gandhi, Abhinav Gupta, and Ruslan Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. arXiv preprint arXiv:2007.00643, 2020.

[77] Bei Chen, Fengji Zhang, Anh Nguyen, Daoguang Zan, Zeqi Lin, Jian-Guang Lou, and Weizhu Chen. Codet: Code generation with generated tests. *arXiv preprint arXiv:2207.10397*, 2022.

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

[79] Dave Zhenyu Chen, Angel X. Chang, and Matthias Nießner. Scanrefer: 3d object localization in rgb-d scans using natural language. arXiv preprint arXiv:1912.08830, 2020.

[80] Haonan Chen, Jingxiang Guo, Bangjun Wang, Tianrui Zhang, Xuchuan Huang, Boren Zheng, Yiwen Hou, Chenrui Tie, Jiajun Deng, and Lin Shao. Goal-VLA: Image-generative vlms as object-centric world models empowering zero-shot robot manipulation. In *2026 IEEE International Conference on Robotics and Automation (ICRA)*, 2026.

[81] Min Chen, Shengjun Zhang, Yuxin Li, Zhang Zhang, Xin Fei, Chong Xia, and Yueqi Duan. Parallelworld: Test-time scaling for embodied reasoning. *arXiv preprint arXiv:2608.22971*, 2026.

[82] Yongchao Chen, Jacob Arkin, Charles Dawson, Yang Zhang, Nicholas Roy, and Chuchu Fan. Autotamp: Autoregressive task and motion planning with llms as translators and checkers. In *2024 IEEE International conference on robotics and automation (ICRA)*, pages 6695–6702. IEEE, 2024.

[83] Zeren Chen, Zhelun Shi, Xiaoya Lu, Lehan He, Sucheng Qian, Zhenfei Yin, Wanli Ouyang, Jing Shao, Yu Qiao, Cewu Lu, et al. RH20T-P: A Primitive-Level Robotic Dataset Towards Composable Generalization Agents. *arXiv preprint arXiv:2403.19622*, 2024.

[84] An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. Spatialrgpt: Grounded spatial reasoning in vision language models. arXiv preprint arXiv:2406.01584, 2024.

[85] Richard Cheng, Gábor Orosz, Richard M. Murray, and Joel W. Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3387–3395, 2019.

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

[87] Ho Jin Choi and Nadia Figueroa. Gaussian process-based active exploration strategies in vision and touch. *arXiv preprint arXiv:2507.05522*, 2025.

[88] Wonje Choi, Woo Kyung Kim, Minjong Yoo, and Honguk Woo. Embodied CoT Distillation From LLM To Off-the-shelf Agents. *arXiv preprint arXiv:2412.11499*, 2024.

[89] Wei Chow, Jiageng Mao, Boyi Li, Daniel Seita, Vitor Guizilini, and Yue Wang. Physbench: Benchmarking and enhancing vision-language models for physical world understanding. arXiv preprint arXiv:2501.16411, 2025.

[90] Ian Chuang, Andrew Lee, Dechen Gao, M-Mahdi Naddaf-Sh, and Iman Soltani. Active vision might be all you need: Exploring active vision in bimanual robotic manipulation. arXiv preprint arXiv:2409.17435, 2025.

[91] Andy Clark and Margaret A Boden. *Being there: Putting brain, body, and world together again*, volume 10. MIT press Cambridge, MA, 1997.

[92] C. Connolly. The determination of next best views. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, page 432–435. Institute of Electrical and Electronics Engineers, 1985.

[93] Yuchen Cui, Siddharth Karamcheti, Raj Palleti, Nidhya Shivakumar, Percy Liang, and Dorsa Sadigh. "No, to the Right" – Online Language Corrections for Robotic Manipulation via Shared Autonomy. *arXiv preprint arXiv:2301.02555*, 2023.

[94] Yinpei Dai, Hongze Fu, Jayjun Lee, Yuejiang Liu, Haoran Zhang, Jianing Yang, Chelsea Finn, Nima Fazeli, and Joyce Chai. Robomme: Benchmarking and understanding memory for robotic generalist policies. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 22396–22450. PMLR, 2026.

[95] Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. In *arXiv preprint arXiv:1801.08757*, 2018.

[96] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Antonino Furnari, Evangelos Kazakos, Jian Ma, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision. *arXiv preprint arXiv:2006.13256*, 2020.

[97] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2135–213509. IEEE, 2018.

[98] Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. RoboNet: Large-Scale Multi-Robot Learning. *arXiv preprint arXiv:1910.11215*, 2019.

[99] Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 91–104. IEEE, 2025.

[100] Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, Dustin Schwenk, Eli VanderBilt, Matthew Wallingford, et al. Robothor: An open simulation-to-real embodied ai platform. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3161–3171. IEEE, 2020.

[101] Xin Ding, Liang Mi, Mingzhe Huang, Zixuan Wang, Chao Zhang, Zixu Hao, Fu Chen, Xiangyu Li, Yikai Zheng, Yaoyu Guo, Weijun Wang, Kun Li, Hao Wu, Yunxin Liu, and Ting Cao. Zetta ζ: An efficient closed-loop embodied harness for self-evolving physical intelligence, 2026.

[102] Mingyu Dong, Chong Xia, Mingyuan Jia, Weichen Lyu, Long Xu, Zheng Zhu, and Yueqi Duan. ReplicateAnyScene: Zero-Shot Video-to-3D Composition via Textual-Visual-Spatial Alignment. *arXiv preprint arXiv:2604.10789*, 2026.

[103] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. PaLM-E: An Embodied Multimodal Language Model. *arXiv preprint arXiv:2303.03378*, 2023.

[104] Yilun Du, Sherry Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B Tenenbaum, Leslie Kaelbling, et al. Video language planning. In *International Conference on Learning Representations*, volume 2024, pages 31138–31155, 2024.

[105] Yuqing Du, Ksenia Konyushkova, Misha Denil, Akhil Raju, Jessica Landon, Felix Hill, Nando de Freitas, and Serkan Cabi. Vision-Language Models as Success Detectors. *arXiv preprint arXiv:2303.07280*, 2023.

[106] Jiafei Duan, Wilbert Pumacay, Nishanth Kumar, Yi Ru Wang, Shulin Tian, Wentao Yuan, Ranjay Krishna, Dieter Fox, Ajay Mandlekar, and Yijie Guo. Aha: A vision-language-model for detecting and reasoning over failures in robotic manipulation. arXiv preprint arXiv:2410.00371, 2024.

[107] Jiafei Duan, Wilbert Pumacay, Nishanth Kumar, Yi Ru Wang, Shulin Tian, Wentao Yuan, Ranjay Krishna, Dieter Fox, Ajay Mandlekar, and Yijie Guo. AHA: A Vision-Language-Model for Detecting and Reasoning Over Failures in Robotic Manipulation. *arXiv preprint arXiv:2410.00371*, 2024.

[108] Frederik Ebert, Chelsea Finn, Alex X. Lee, and Sergey Levine. Self-Supervised Visual Planning with Temporal Skip Connections. *arXiv preprint arXiv:1710.05268*, 2017.

[109] Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Chenxi Wang, Junbo Wang, Haoyi Zhu, and Cewu Lu. RH20T: A Comprehensive Robotic Dataset for Learning Diverse Skills in One-Shot. *arXiv preprint arXiv:2307.00595*, 2023.

[110] Hao-Shu Fang, Chenxi Wang, Minghao Gou, and Cewu Lu. Graspnet-1billion: A large-scale benchmark for general object grasping. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11441–11450. IEEE, 2020.

[111] Xin Fei, Zhixuan Xu, Huaicong Fang, Tianrui Zhang, and Lin Shao. T(R,O) grasp: Efficient graph diffusion of robot-object spatial transformation for cross-embodiment dexterous grasping. In *2026 IEEE International Conference on Robotics and Automation (ICRA)*, 2026.

[112] Yuchun Feng, Jinliang Zheng, Zhihao Wang, Dongxiu Liu, Jianxiong Li, Jiangmiao Pang, Tai Wang, and Xianyuan Zhan. Demystifying action space design for robotic manipulation policies. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 30642–30663. PMLR, 2026.

[113] Yunhai Feng, Jiaming Han, Zhuoran Yang, Xiangyu Yue, Sergey Levine, and Jianlan Luo. Reflective planning: Vision-language models for multi-stage long-horizon robotic manipulation. *arXiv preprint arXiv:2502.16707*, 2025.

[114] Zhiyuan Feng, Zhaolu Kang, Qijie Wang, Zhiying Du, Jiongrui Yan, Shubin Shi, Chengbo Yuan, Huizhi Liang, Yu Deng, Qixiu Li, Rushuai Yang, Arctanx An, Leqi Zheng, Weijie Wang, Shawn Chen, Sicheng Xu, Yaobo Liang, Jiaolong Yang, and Baining Guo. Seeing across views: Benchmarking spatial reasoning of vision-language models in robotic scenes. arXiv preprint arXiv:2510.19400, 2026.

[115] Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1515–1528. PMLR, 2018.

[116] Letian Fu, Justin Yu, Karim El-Refai, Ethan Kou, Haoru Xue, Huang Huang, Wenli Xiao, Li Fei-Fei, Guanya Shi, Jiajun Wu, S. Shankar Sastry, Yuke Zhu, Ken Goldberg, and Linxi Fan. Cap-x: A framework for benchmarking and improving coding agents for robot manipulation. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 32155–32212. PMLR, 2026.

[117] Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A. Smith, Wei-Chiu Ma, and Ranjay Krishna. Blink: Multimodal large language models can see but not perceive. arXiv preprint arXiv:2404.12390, 2024.

[118] Zipeng Fu, Tony Z. Zhao, and Chelsea Finn. Mobile ALOHA: Learning Bimanual Mobile Manipulation with Low-Cost Whole-Body Teleoperation. *arXiv preprint arXiv:2401.02117*, 2024.

[119] Jensen Gao, Bidipta Sarkar, Fei Xia, Ted Xiao, Jiajun Wu, Brian Ichter, Anirudha Majumdar, and Dorsa Sadigh. Physically grounded vision-language models for robotic manipulation. arXiv preprint arXiv:2309.02561, 2024.

[120] Javier García and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(42):1437–1480, 2015.

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

[122] Haoran Geng, Feishi Wang, Songlin Wei, Yuyang Li, Bangjun Wang, Boshi An, Haozhe Lou, Charlie Tianyue Cheng, Peihao Li, Haozhe Chen, Yutong Liang, Yuxi Qian, Jiageng Mao, Weikang Wan, Yiran Geng, Mingtong Zhang, Jiangran Lyu, Siheng Zhao, Jiazhao Zhang, Chaoyi Xu, Jialiang Zhang, Chengyang Zhao, Haoran Lu, Yufei Ding, Ran Gong, Yuran Wang, Yuxuan Kuang, Ruihai Wu, Baoxiong Jia, Hao Dong, Siyuan Huang, Yue Wang, Jitendra Malik, and Pieter Abbeel. Roboverse: A unified platform, benchmark and dataset for scalable and generalizable robot learning. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, 2025.

[123] James J Gibson. *The ecological approach to visual perception: classic edition*. Psychology press, 2014.

[124] Anand Gokhale, Vaibhav Srivastava, and Francesco Bullo. LogicGuard: Improving Embodied LLM Agents through Temporal Logic Based Critics. *arXiv preprint arXiv:2507.03293*, 2025.

[125] Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: Large Language Models Can Self-Correct with Tool-Interactive Critiquing. In *International Conference on Learning Representations*, 2024.

[126] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4D: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[127] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, et al. Ego-Exo4D: Understanding skilled human activity from first- and third-person perspectives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[128] Qiao Gu, Alihusein Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, et al. ConceptGraphs: Open-Vocabulary 3D Scene Graphs for Perception and Planning. *arXiv preprint arXiv:2309.16650*, 2023.

[129] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary object detection via vision and language knowledge distillation. *arXiv preprint arXiv:2104.13921*, 2021.

[130] Atharva Gundawar, Som Sagar, and Ransalu Senanayake. Pac bench: Do foundation models understand prerequisites for executing manipulation policies? In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[131] Agrim Gupta, Piotr Dollár, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. arXiv preprint arXiv:1908.03195, 2019.

[132] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. arXiv preprint arXiv:2211.11559, 2022.

[133] Gary S Guthart and J Kenneth Salisbury. The intuitive/sup tm/telesurgery system: overview and application. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 1, pages 618–621. IEEE, 2000.

[134] Sami Haddadin, Alin Albu-Schäffer, and Gerd Hirzinger. Requirements for safe robots: Measurements, analysis and new insights. *The International Journal of Robotics Research*, 28(11–12):1507–1527, 2009.

[135] Songhao Han, Boxiang Qiu, Yue Liao, Siyuan Huang, Chen Gao, Shuicheng Yan, and Si Liu. Robocerebra: A large-scale benchmark for long-horizon robotic manipulation evaluation. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[136] Xinyi He, Ying Yang, Chuanjian Fu, Sihan Guo, Songchun Zhu, Lifeng Fan, Zhenliang Zhang, and Yujia Peng. Automatic cognitive task generation for in-situ evaluation of embodied agents. arXiv preprint arXiv:2602.05249, 2026.

[137] Yi-Xiang He, Lan Wei, Haoming Cen, Jian-Jian Jiang, Zhuohao Li, Guanxing Lu, Yihan Yang, Dandan Zhang, and Wei-Shi Zheng. A closed-loop multi-agent framework for robust multi-robot manipulation. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026.

[138] Carolina Higuera, Akash Sharma, Chaithanya Krishna Bodduluri, Taosha Fan, Patrick Lancaster, Mrinal Kalakrishnan, Michael Kaess, Byron Boots, Mike Lambeta, Tingfan Wu, et al. Sparsh: Self-supervised touch representations for vision-based tactile sensing. *arXiv preprint arXiv:2410.24090*, 2024.

[139] Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. *Advances in Neural Information Processing Systems*, 36:20482–20494, 2023.

[140] Yining Hong, Zishuo Zheng, Peihao Chen, Yian Wang, Junyan Li, and Chuang Gan. Multiply: A multisensory object-centric embodied large language model in 3d world. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26396–26406. IEEE, 2024.

[141] Ryan Hoque, Peide Huang, David J. Yoon, Mouli Sivapurapu, and Jian Zhang. EgoDex: Learning Dexterous Manipulation from Large-Scale Egocentric Video. *arXiv preprint arXiv:2505.11709*, 2025.

[142] Ryan Hoque, Ajay Mandlekar, Caelan Garrett, Ken Goldberg, and Dieter Fox. IntervenGen: Interventional Data Generation for Robust and Data-Efficient Robot Imitation Learning. *arXiv preprint arXiv:2405.01472*, 2024.

[143] Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. VIME: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems*, volume 29, pages 1109–1117, 2016.

[144] Mengkang Hu, Yao Mu, Xinmiao Yu, Mingyu Ding, Shiguang Wu, Wenqi Shao, Qiguang Chen, Bin Wang, Yu Qiao, and Ping Luo. Tree-planner: Efficient close-loop task planning with large language models. In *International Conference on Learning Representations*, volume 2024, pages 48669–48696, 2024.

[145] Mu Hu, Wei Yin, Chi Zhang, Zhipeng Cai, Xiaoxiao Long, Kaixuan Wang, Hao Chen, Gang Yu, Chunhua Shen, and Shaojie Shen. Metric3dv2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. arXiv preprint arXiv:2404.15506, 2025.

[146] Songqiao Hu, Zeyi Liu, Shuang Liu, Jun Cen, Zihan Meng, Shihefeng Wang, Xiang Li, and Xiao He. Vlsa: Vision-language-action models with plug-and-play safety constraint layer. arXiv preprint arXiv:2512.11891, 2025.

[147] Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. *arXiv preprint arXiv:2412.14803*, 2024.

[148] Zheyuan Hu, Robyn Wu, Naveen Enock, Jasmine Li, Riya Kadakia, Zackory Erickson, and Aviral Kumar. RaC: Robot Learning for Long-Horizon Tasks by Scaling Recovery and Correction. *arXiv preprint arXiv:2509.07953*, 2025.

[149] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10608–10615. IEEE, 2023.

[150] Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large Language Models Cannot Self-Correct Reasoning Yet. In *International Conference on Learning Representations*, 2024.

[151] Jinbang Huang, Yuanzhao Hu, Zhiyuan Li, Ran Qi, Yixin Xiao, Zhanguang Zhang, Mark Coates, Tongtong Cao, and Yingxue Zhang. Roboharness: Memory-driven orchestration of heterogeneous robot policies for long-horizon planning. *arXiv preprint arXiv:2607.18060*, 2026.

[152] Mu Huang, Hui Wang, Kerui Ren, Linning Xu, Yunsong Zhou, Mulin Yu, Bo Dai, and Jiangmiao Pang. Soma: A real-to-sim neural simulator for robotic soft-body manipulation. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 47834–47850. PMLR, 2026.

[153] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024.

[154] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[155] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[156] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner Monologue: Embodied Reasoning through Planning with Language Models. *arXiv preprint arXiv:2207.05608*, 2022.

[157] Nathan Hughes, Yun Chang, and Luca Carlone. Hydra: A real-time spatial perception system for 3d scene graph construction and optimization. *arXiv preprint arXiv:2201.13360*, 2022.

[158] Auke Jan Ijspeert, Jun Nakanishi, Heiko Hoffmann, Peter Pastor, and Stefan Schaal. Dynamical movement primitives: learning attractor models for motor behaviors. *Neural computation*, 25(2):328–373, 2013.

[159] Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Greg Balke, Kevin Black, George Bokinsky, Shihao Cao, Thomas Charbonnier, et al. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities. *arXiv preprint arXiv:2604.15483*, 2026.

[160] Stefan Isler, Reza Sabzevari, Jeffrey Delmerico, and Davide Scaramuzza. An information gain formulation for active volumetric 3d reconstruction. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, page 3477–3484. IEEE, May 2016.

[161] Klemens Iten, Alexander Proshkin, Bhavya Sukhija, Stelian Coros, Andreas Krause, Pieter Abbeel, and Carmelo Sferrazza. Tactile curiosity drives robot interaction. *arXiv preprint arXiv:2609.40134*, 2026.

[162] Arhan Jain, Mingtong Zhang, Kanav Arora, William Chen, Marcel Torne, Muhammad Zubair Irshad, Sergey Zakharov, Yue Wang, Sergey Levine, Chelsea Finn, Wei-Chiu Ma, Dhruv Shah, Abhishek Gupta, and Karl Pertsch. Polaris: Scalable real-to-sim evaluations for generalist robot policies. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026.

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

[164] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. BC-Z: Zero-shot task generalization with robotic imitation learning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 991–1002. PMLR, 2022.

[165] Yash Jangir, Yidi Zhang, Kashu Yamazaki, Chenyu Zhang, Kuan-Hsun Tu, Tsung-Wei Ke, Lei Ke, Yonatan Bisk, and Katerina Fragkiadaki. Robotarena ∞: Scalable robot benchmarking via real-to-sim translation. In *International Conference on Learning Representations*, 2026.

[166] Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, Mohd Omama, Tao Chen, Alaa Maalouf, Shuang Li, Ganesh Iyer, Soroush Saryazdi, Nikhil Keetha, et al. Conceptfusion: Open-set multimodal 3d mapping. *arXiv preprint arXiv:2302.07241*, 2023.

[167] Ahad Jawaid and Yu Xiang. OpenEgo: A large-scale multimodal egocentric dataset for dexterous manipulation. *arXiv preprint arXiv:2509.05513*, 2025.

[168] Dinesh Jayaraman and Kristen Grauman. Learning to look around: Intelligently exploring unseen environments for unknown tasks. arXiv preprint arXiv:1709.00507, 2017.

[169] Yuheng Ji, Huajie Tan, Jiayu Shi, Xiaoshuai Hao, Yuan Zhang, Hengyuan Zhang, Pengwei Wang, Mengdi Zhao, Yao Mu, Pengju An, et al. RoboBrain: A Unified Brain Model for Robotic Manipulation from Abstract to Concrete. *arXiv preprint arXiv:2502.21257*, 2025.

[170] Yuheng Ji, Huajie Tan, Jiayu Shi, Xiaoshuai Hao, Yuan Zhang, Hengyuan Zhang, Pengwei Wang, Mengdi Zhao, Yao Mu, Pengju An, Xinda Xue, Qinghang Su, Huaihai Lyu, Xiaolong Zheng, Jiaming Liu, Zhongyuan Wang, and Shanghang Zhang. Robobrain: A unified brain model for robotic manipulation from abstract to concrete. arXiv preprint arXiv:2502.21257, 2025.

[171] Baoyang Jiang, Fengchun Zhang, Leyuan Wang, Haotian Li, Yida Wang, Zhe Ji, Jinshan Lai, Xi Ren, Danyang Li, Zheng Yang, Jianwei Hu, and Qiang Ma. Embodied-BenchForge: A closed-loop agentic workflow for embodied benchmark construction. arXiv preprint arXiv:2609.13082, 2026.

[172] Hai Jiang, Yixian Zou, Binbin Liang, Boqian Liu, Fanman Meng, and Shuaicheng Liu. FutureRTC: Real-time robot execution with anticipatory-conditioned action chunking. *arXiv preprint arXiv:2607.24008*, 2026.

[173] Hanxiao Jiang, Binghao Huang, Ruihai Wu, Zhuoran Li, Shubham Garg, Hooshang Nayyeri, Shenlong Wang, and Yunzhu Li. Roboexp: Action-conditioned scene graph via interactive exploration for robotic manipulation. arXiv preprint arXiv:2402.15487, 2024.

[174] Jing Jiang, Yue Yang, Xinkai Jiang, Gedas Bertasius, Daniel J. Szafir, and Rudolf Lioutikov. From rollout to reset: A graph-based harness for autonomous long-horizon manipulation evaluation. arXiv preprint arXiv:2609.19413, 2026.

[175] Qing Jiang, Xingyu Chen, Zhaoyang Zeng, Junzhi Yu, and Lei Zhang. Rex-thinker: Grounded object referring via chain-of-thought reasoning. arXiv preprint arXiv:2506.04034, 2025.

[176] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. arXiv preprint arXiv:2202.08227, 2022.

[177] Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can Language Models Resolve Real-World GitHub Issues? In *International Conference on Learning Representations*, 2024.

[178] Piaopiao Jin, Qi Wang, Guokang Sun, Ziwen Cai, Pinjia He, and Yangwei You. Dual-Actor Fine-Tuning of VLA Models: A Talk-and-Tweak Human-in-the-Loop Approach. *arXiv preprint arXiv:2509.13774*, 2025.

[179] Joshua Jones, Oier Mees, Carmelo Sferrazza, Kyle Stachowicz, Pieter Abbeel, and Sergey Levine. Beyond sight: Finetuning generalist robot policies with heterogeneous sensors via language grounding. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5961–5968. IEEE, 2025.

[180] Yuanchen Ju, Kaizhe Hu, Guowei Zhang, Gu Zhang, Mingrun Jiang, and Huazhe Xu. Robo-abc: Affordance generalization beyond categories via semantic correspondence for robot manipulation. In *European Conference on Computer Vision*, pages 222–239. Springer, 2024.

[181] Yuanchen Ju, Yongyuan Liang, Yen-Jen Wang, Nandiraju Gireesh, Yuanliang Ju, Seungjae Lee, Qiao Gu, Elvis Hsieh, Furong Huang, and Koushil Sreenath. Momagraph: State-aware unified scene graphs with vision-language model for embodied task planning. arXiv preprint arXiv:2512.16909, 2026.

[182] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on robot learning*, pages 651–673. PMLR, 2018.

[183] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. Mdetr-modulated detection for end-to-end multi-modal understanding. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pages 1760–1770. IEEE, 2021.

[184] Amita Kamath, Jack Hessel, and Kai-Wei Chang. What's "up" with vision-language models? investigating their struggle with spatial reasoning. arXiv preprint arXiv:2310.19785, 2023.

[185] Xuhui Kang, Sung-Wook Lee, Haolin Liu, Yuyan Wang, and Yen-Ling Kuo. Moving out: Physically-grounded human-ai collaboration. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 55597–55629. PMLR, 2026.

[186] Yiping Kang, Johann Hauswald, Cao Gao, Austin Rovinski, Trevor Mudge, Jason Mars, and Lingjia Duan. Neurosurgeon: Collaborative intelligence between the cloud and mobile edge. In *ACM ASPLOS*, 2017.

[187] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. arXiv preprint arXiv:2307.07635, 2024.

[188] Nikhil Keetha, Norman Müller, Johannes Schönberger, Lorenzo Porzi, Yuchen Zhang, Tobias Fischer, Arno Knapitsch, Duncan Zauss, Ethan Weber, Nelson Antunes, Jonathon Luiten, Manuel Lopez-Antequera, Samuel Rota Bulò, Christian Richardt, Deva Ramanan, Sebastian Scherer, and Peter Kontschieder. Mapanything: Universal feed-forward metric 3d reconstruction. arXiv preprint arXiv:2509.13414, 2026.

[189] Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J. Kochenderfer. HG-DAgger: Interactive Imitation Learning with Human Experts, 2018.

[190] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19672–19682. IEEE, 2023.

[191] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. DROID: A Large-Scale In-The-Wild Robot Manipulation Dataset. *arXiv preprint arXiv:2403.12945*, 2024.

[192] Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. *arXiv preprint arXiv:2210.02406*, 2022.

[193] Dabin Kim, Daemin Park, Sangyub Lee, Jinsik Kim, Yeongtak Oh, Jongho Shin, and Sungroh Yoon. Safe embodied AI for long-horizon tasks: A cross-layer analysis of robotic manipulation. arXiv preprint arXiv:2606.05660, 2026.

[194] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *Robotics: Science and Systems (RSS)*, 2025.

[195] Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, et al. Cosmos policy: Fine-tuning video models for visuomotor control and planning. *arXiv preprint arXiv:2601.16163*, 2026.

[196] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model, 2024. *URL https://arxiv. org/abs/2406.09246*, 1(2):4, 2024.

[197] Yejin Kim, Wilbert Pumacay, Omar Rayyan, Max Argus, Winson Han, Eli Vanderbilt, Jordi Salvador, Abhay Deshpande, Rose Hendrix, Snehal Jauhri, Shuo Liu, Nur Muhammad Mahi Shafiullah, Maya Guru, Ainaz Eftekhar, Karen Farley, Donovan Clay, Jiafei Duan, Arjun Guru, Piper Wolters, Alvaro Herrasti, Ying-Chun Lee, Georgia Chalvatzaki, Yuchen Cui, Ali Farhadi, Dieter Fox, and Ranjay Krishna. Molmospaces: Large-scale open ecosystem for robot manipulation and navigation. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026.

[198] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[199] Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[200] Jens Kober and Jan Peters. Policy search for motor primitives in robotics. *Advances in neural information processing systems*, 21, 2008.

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

[202] James Kuffner. Cloud-enabled robots. In *IEEE-RAS Int'l Conf. on Humanoid Robots*, 2010.

[203] James J Kuffner and Steven M LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, volume 2, pages 995–1001. IEEE, 2000.

[204] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *ACM SOSP*, 2023.

[205] Yann Labbé, Lucas Manuelli, Arsalan Mousavian, Stephen Tyree, Stan Birchfield, Jonathan Tremblay, Justin Carpentier, Mathieu Aubry, Dieter Fox, and Josef Sivic. Megapose: 6d pose estimation of novel objects via render & compare. arXiv preprint arXiv:2212.06870, 2022.

[206] Bolin Lai, Fiona Ryan, Wenqi Jia, Miao Liu, and James M. Rehg. Listen to look into the future: Audio-visual egocentric gaze anticipation. *arXiv preprint arXiv:2305.03907*, 2023.

[207] Jinshan Lai, Jianwei Hu, Baoyang Jiang, Fengchun Zhang, Leyuan Wang, Haotian Li, Yida Wang, Tingxuan Huang, Xi Ren, and Qiang Ma. Intelligent automation for embodied benchmark construction: Pipelines, embodiments, simulators, and trends. arXiv preprint arXiv:2606.12207, 2026.

[208] Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. Lisa: Reasoning segmentation via large language model. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9579–9589. IEEE, 2024.

[209] George Lakoff, Mark Johnson, and John F Sowa. Review of philosophy in the flesh: The embodied mind and its challenge to western thought. *Computational Linguistics*, 25(4):631–634, 1999.

[210] Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, Winson Han, Wilbert Pumacay, Angelica Wu, Rose Hendrix, Karen Farley, Eli VanderBilt, Ali Farhadi, Dieter Fox, and Ranjay Krishna. Molmoact: Action reasoning models that can reason in space. arXiv preprint arXiv:2508.07917, 2025.

[211] Sangoh Lee, Sangwoo Mo, and Wook-Shin Han. Bring my cup! personalizing vision-language-action models with visual attentive prompting. arXiv preprint arXiv:2512.20014, 2026.

[212] Yumin Lee, Hyoseok Ju, and Giseop Kim. Lt-mem: Volatility-aware spatio-temporal memory for lifelong scene understanding. arXiv preprint arXiv:2608.19059, 2026.

[213] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. arXiv preprint arXiv:2406.09756, 2024.

[214] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016.

[215] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International journal of robotics research*, 37(4-5):421–436, 2018.

[216] Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, Mona Anvari, Minjune Hwang, Manasi Sharma, Arman Aydin, Dhruva Bansal, Samuel Hunter, Kyu-Young Kim, Alan Lou, Caleb R. Matthews, Ivan Villa-Renteria, Jerry Huayang Tang, Claire Tang, Fei Xia, Silvio Savarese, Hyowon Gweon, Karen Liu, Jiajun Wu, and Li Fei-Fei. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 80–93. PMLR, 2023.

[217] Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial forcing: Implicit spatial representation alignment for vision-language-action model. arXiv preprint arXiv:2510.12276, 2025.

[218] Hongyu Li, Mingxi Jia, Tuluhan Akbulut, Yu Xiang, George Konidaris, and Srinath Sridhar. V-hop: Visuo-haptic 6d object pose tracking. arXiv preprint arXiv:2502.17434, 2025.

[219] Jan-Mou Li, Paul Schmitt, Wei Tong, Majed Mohammed, Akshay Chalana, Arpan Kusari, and Edward Griffor. Embodied AI in action: Insights from SAE world congress 2026 on safety, trust, robotics, and real-world deployment. arXiv preprint arXiv:2605.10653, 2026.

[220] Jierui Li, Hung Le, Yingbo Zhou, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Codetree: Agent-guided tree search for code generation with large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3711–3726, 2025.

[221] Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026.

[222] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10955–10965. IEEE, 2022.

[223] Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Li E Li, Ruohan Zhang, et al. Embodied agent interface: Benchmarking llms for embodied decision making. *Advances in Neural Information Processing Systems*, 37:100428–100534, 2024.

[224] Pengfei Li, Beiwen Tian, Yongliang Shi, Xiaoxue Chen, Hao Zhao, Guyue Zhou, and Ya-Qin Zhang. Toist: Task oriented instance segmentation transformer with noun-pronoun distillation. *Advances in Neural Information Processing Systems*, 35:17597–17611, 2022.

[225] Siyuan Li, Taiyan Lang, Aoqi Yan, Jia Yu, Feifan Liu, Yihan Du, Yu Zheng, Xun Wang, and Peng Liu. Safe task planning with long-term graph memory for embodied agents. arXiv preprint arXiv:2609.08444, 2026.

[226] Xiao Li, Xiang Zheng, Yifeng Gao, Xinyu Xia, Yixu Wang, Xin Wang, Ye Sun, Yunhan Zhao, Ming Wen, Jiayu Li, et al. Safety in embodied ai: A survey of risks, attacks, and defenses. *arXiv preprint arXiv:2605.02900*, 2026.

[227] Xinghang Li, Jun Guo, Qiwei Li, Long Qian, Hang Lai, Yueze Wang, Hongyu Yan, Jiahang Cao, Xi Chen, Jingen Qu, et al. Xiaomi-robotics-u0: Unified embodied synthesis with world foundation model. *arXiv preprint arXiv:2607.11643*, 2026.

[228] Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, Hang Li, and Tao Kong. Vision-language foundation models as effective robot imitators. In *International Conference on Learning Representations (ICLR)*, 2024.

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

[230] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pages 292–305, 2023.

[231] Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022.

[232] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B. Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. arXiv preprint arXiv:1810.01566, 2019.

[233] Yuyang Li, Wenxin Du, Chang Yu, Puhao Li, Zihang Zhao, Tengyu Liu, Chenfanfu Jiang, Yixin Zhu, and Siyuan Huang. Taccel: Scaling up vision-based tactile robotics via high-performance gpu simulation. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[234] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as Policies: Language Model Programs for Embodied Control. *arXiv preprint arXiv:2209.07753*, 2022.

[235] Chunru Lin, Haotian Yuan, Yian Wang, Xiaowen Qiu, Tsun-Hsuan Johnson Wang, Minghao Guo, Bohan Wang, Yashraj Narang, Dieter Fox, and Chuang Gan. Robotsmith: Generative robotic tool design for acquisition of complex manipulation skills. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[236] Kevin Lin, Christopher Agia, Toki Migimatsu, Marco Pavone, and Jeannette Bohg. Text2motion: From natural language instructions to feasible plans. *Autonomous Robots*, 47(8):1345–1365, 2023.

[237] Bo Liu, Yuqian Jiang, Xiaohan Zhang, Qiang Liu, Shiqi Zhang, Joydeep Biswas, and Peter Stone. Llm+ p: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*, 2023.

[238] Chang Liu, Henghui Ding, and Xudong Jiang. Gres: Generalized referring expression segmentation. arXiv preprint arXiv:2306.00968, 2023.

[239] Cuiyu Liu, Wei Zhai, Yuhang Yang, Hongchen Luo, Sen Liang, Yang Cao, and Zheng-Jun Zha. Grounding 3d scene affordance from egocentric interactions. *arXiv preprint arXiv:2409.19650*, 2024.

[240] Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, Zhan Ling, Fatih Porikli, and Hao Su. Partslip: Low-shot part segmentation for 3d point clouds via pretrained image-language models. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21736–21746. IEEE, 2023.

[241] Peiqi Liu, Zhanqiu Guo, Mohit Warke, Soumith Chintala, Chris Paxton, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Dynamem: Online dynamic spatio-semantic memory for open world mobile manipulation. arXiv preprint arXiv:2411.04999, 2025.

[242] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*, pages 38–55. Springer, 2024.

[243] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1B: a diffusion foundation model for bimanual manipulation. In *International Conference on Learning Representations (ICLR)*, 2025.

[244] Yuchen Liu, Luigi Palmieri, Sebastian Koch, Ilche Georgievski, and Marco Aiello. Delta: Decomposed efficient long-term robot task planning using large language models. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10995–11001. IEEE, 2025.

[245] Yuejiang Liu, Alexandre Alahi, Chris Russell, Max Horn, Dominik Zietlow, Bernhard Schölkopf, and Francesco Locatello. Causal Triplet: An Open Challenge for Intervention-centric Causal Representation Learning. *arXiv preprint arXiv:2301.05169*, 2023.

[246] Yuhong Liu, Beichen Zhang, Yuhang Zang, Yuhang Cao, Long Xing, Xiaoyi Dong, Haodong Duan, Dahua Lin, and Jiaqi Wang. Spatial-ssrl: Enhancing spatial understanding via self-supervised reinforcement learning. arXiv preprint arXiv:2510.27606, 2025.

[247] Zeyi Liu, Arpit Bahety, and Shuran Song. REFLECT: Summarizing Robot Experiences for Failure Explanation and Correction. In *Conference on Robot Learning*, 2023.

[248] Xiaoya Lu, Zeren Chen, Xuhao Hu, Yijin Zhou, Weichen Zhang, Dongrui Liu, Lu Sheng, and Jing Shao. Is-bench: Evaluating interactive safety of vlm-driven embodied agents in daily household tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 35680–35688, 2026.

[249] Xiaoya Lu, Yijin Zhou, Zeren Chen, Ruocheng Wang, Bingrui Sima, Enshen Zhou, Lu Sheng, Dongrui Liu, and Jing Shao. Homeguard: Vlm-based embodied safeguard for identifying contextual risk in household task. Lecture Notes in Computer Science, 2026.

[250] Max Lungarella, Giorgio Metta, Rolf Pfeifer, and Giulio Sandini. Developmental robotics: a survey. *Connection science*, 15(4):151–190, 2003.

[251] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 1113–1132. PMLR, 2020.

[252] Weichen Lyu, Chong Xia, Mingyu Dong, Long Xu, Shengjun Zhang, and Yueqi Duan. Physga: A physics-space grounded agentic framework for generalizable and interactive 3d composition, 2026.

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

[254] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-Level Reward Design via Coding Large Language Models. In *International Conference on Learning Representations*, 2024.

[255] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-Refine: Iterative Refinement with Self-Feedback. *arXiv preprint arXiv:2303.17651*, 2023.

[256] Dominic Maggio, Yun Chang, Nathan Hughes, Matthew Trang, Dan Griffith, Carlyn Dougherty, Eric Cristofalo, Lukas Schmid, and Luca Carlone. Clio: Real-time task-driven open-set 3d scene graphs. *IEEE Robotics and Automation Letters*, 9(10):8921–8928, 2024.

[257] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*, 2017.

[258] 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, Alexander Sax, and Aravind Rajeswaran. Openeqa: Embodied question answering in the era of foundation models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 16488–16498. IEEE, 2024.

[259] Arjun Majumdar, Karmesh Yadav, Sergio Arnaud, Yecheng Jason Ma, Claire Chen, Sneha Silwal, Aryan Jain, Vincent-Pierre Berges, Pieter Abbeel, Jitendra Malik, Dhruv Batra, Yixin Lin, Oleksandr Maksymets, Aravind Rajeswaran, and Franziska Meier. Where are we in the search for an artificial visual cortex for embodied intelligence? arXiv preprint arXiv:2303.18240, 2024.

[260] Yunze Man, Liang-Yan Gui, and Yu-Xiong Wang. Situational awareness matters in 3d vision language reasoning. arXiv preprint arXiv:2406.07544, 2024.

[261] Zhao Mandi, Shreeya Jain, and Shuran Song. Roco: Dialectic multi-robot collaboration with large language models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 286–299. IEEE, 2024.

[262] Ajay Mandlekar, Jonathan Booher, Max Spero, Albert Tung, Anchit Gupta, Yuke Zhu, Animesh Garg, Silvio Savarese, and Li Fei-Fei. Scaling Robot Supervision to Hundreds of Hours with RoboTurk: Robotic Manipulation Dataset through Human Reasoning and Dexterity. *arXiv preprint arXiv:1911.04052*, 2019.

[263] Ajay Mandlekar, Danfei Xu, Roberto Martín-Martín, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. Human-in-the-Loop Imitation Learning using Remote Teleoperation. *arXiv preprint arXiv:2012.06733*, 2020.

[264] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What Matters in Learning from Offline Human Demonstrations for Robot Manipulation. In *Conference on Robot Learning*, 2021.

[265] Joao Marcos Correia Marques, Nils Dengler, Tobias Zaenker, Jesper Mucke, Shenlong Wang, Maren Bennewitz, and Kris Hauser. Map space belief prediction for manipulation-enhanced mapping. arXiv preprint arXiv:2502.20606, 2025.

[266] Damiano Marsili, Rohun Agrawal, Yisong Yue, and Georgia Gkioxari. Visual agentic ai for spatial reasoning with a dynamic api. arXiv preprint arXiv:2502.06787, 2025.

[267] Oier Mees, Lukas Hermann, and Wolfram Burgard. What matters in language conditioned robotic imitation learning over unstructured data. *IEEE Robotics and Automation Letters*, 7(4):11205–11212, 2022.

[268] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, et al. Simple open-vocabulary object detection. In *European conference on computer vision*, pages 728–755. Springer, 2022.

[269] Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Muñoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, et al. Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*, 2025.

[270] Kaichun Mo, Leonidas Guibas, Mustafa Mukadam, Abhinav Gupta, and Shubham Tulsiani. Where2act: From pixels to actions for articulated 3d objects. arXiv preprint arXiv:2101.02692, 2021.

[271] Luis Montesano, Manuel Lopes, Alexandre Bernardino, and José Santos-Victor. Learning Object Affordances: From Sensory–Motor Coordination to Imitation. *IEEE Transactions on Robotics*, 24(1):15–26, 2008.

[272] Hans Moravec. *Mind children: The future of robot and human intelligence*. Harvard University Press, 1988.

[273] Yao Mu, Tianxing Chen, Zanxin Chen, Shijia Peng, Zhiqian Lan, Zeyu Gao, Zhixuan Liang, Qiaojun Yu, Yude Zou, Mingkun Xu, Lunkai Lin, Zhiqiang Xie, Mingyu Ding, and Ping Luo. RoboTwin: Dual-Arm Robot Benchmark with Generative Digital Twins. *arXiv preprint arXiv:2504.13059*, 2025.

[274] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. arXiv preprint arXiv:2203.12601, 2022.

[275] Soroush Nasiriany, Sep Nasiriany, Abhiram Maddukuri, and Yuke Zhu. Robocasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *International Conference on Learning Representations*, 2026.

[276] Phat Nguyen, Tsun-Hsuan Wang, Zhang-Wei Hong, Erfan Aasi, Andrew Silva, Guy Rosman, Sertac Karaman, and Daniela Rus. ReGen: Generative robot simulation via inverse design. arXiv preprint arXiv:2511.04769, 2025.

[277] NVIDIA, :, Alisson Azzolini, Junjie Bai, Hannah Brandon, Jiaxin Cao, Prithvijit Chattopadhyay, Huayu Chen, Jinju Chu, Yin Cui, Jenna Diamond, Yifan Ding, Liang Feng, Francesco Ferroni, Rama Govindaraju, Jinwei Gu, Siddharth Gururani, Imad El Hanafi, Zekun Hao, Jacob Huffman, Jingyi Jin, Brendan Johnson, Rizwan Khan, George Kurian, Elena Lantz, Nayeon Lee, Zhaoshuo Li, Xuan Li, Maosheng Liao, Tsung-Yi Lin, Yen-Chen Lin, Ming-Yu Liu, Xiangyu Lu, Alice Luo, Andrew Mathau, Yun Ni, Lindsey Pavao, Wei Ping, David W. Romero, Misha Smelyanskiy, Shuran Song, Lyne Tchapmi, Andrew Z. Wang, Boxin Wang, Haoxiang Wang, Fangyin Wei, Jiashu Xu, Yao Xu, Dinghao Yang, Xiaodong Yang, Zhuolin Yang, Jingxu Zhang, Xiaohui Zeng, and Zhe Zhang. Cosmos-reason1: From physical common sense to embodied reasoning. arXiv preprint arXiv:2503.15558, 2025.

[278] NVIDIA. GR00T N1: An open foundation model for generalist humanoid robots, 2025. arXiv:2503.14734.

[279] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An Open-Source Generalist Robot Policy. *arXiv preprint arXiv:2405.12213*, 2024.

[280] Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, et al. Open X-Embodiment: Robotic Learning Datasets and RT-X Models. *arXiv preprint arXiv:2310.08864*, 2023.

[281] OpenAI. Gpt-6 astra. https://openai.com/zh-Hans-CN/index/gpt-6-astra/, 2026.

[282] Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.

[283] Jonas Pai, Liam Achenbach, Victoriano Montesinos, Benedek Forrai, Oier Mees, and Elvis Nava. mimic-video: Video-action models for generalizable robot control beyond vlas. *arXiv preprint arXiv:2512.15692*, 2025.

[284] Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[285] Baoqi Pei, Yifei Huang, Jilan Xu, Yuping He, Guo Chen, Fei Wu, Yu Qiao, and Jiangmiao Pang. EgoThinker: Unveiling egocentric reasoning with spatio-temporal CoT. *arXiv preprint arXiv:2510.23569*, 2025.

[286] Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. Kosmos-2: Grounding multimodal large language models to the world. *arXiv preprint arXiv:2306.14824*, 2023.

[287] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: Efficient action tokenization for vision-language-action models. In *Robotics: Science and Systems (RSS)*, 2025.

[288] Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu. Unidepth: Universal monocular metric depth estimation. arXiv preprint arXiv:2403.18913, 2024.

[289] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 3406–3413. IEEE, 2016.

[290] Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. Teacher algorithms for curriculum learning of deep RL in continuously parameterized environments. *arXiv preprint arXiv:1910.07224*, 2019.

[291] Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. Adapt: As-needed decomposition and planning with language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4226–4252, 2024.

[292] Li Puyin, Tiange Xiang, Ella Mao, Shirley Wei, Xinye Chen, Adnan Masood, Li Fei-fei, and Ehsan Adeli. Quantiphy: A quantitative benchmark evaluating physical reasoning abilities of vision-language models. arXiv preprint arXiv:2512.19526, 2025.

[293] Zekun Qi, Wenyao Zhang, Yufei Ding, Runpei Dong, Xinqiang Yu, Jingwen Li, Lingyun Xu, Baoyu Li, Xialin He, Guofan Fan, Jiazhao Zhang, Jiawei He, Jiayuan Gu, Xin Jin, Kaisheng Ma, Zhizheng Zhang, He Wang, and Li Yi. Sofar: Language-grounded orientation bridges spatial reasoning and object manipulation. arXiv preprint arXiv:2502.13143, 2025.

[294] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20051–20060. IEEE, 2024.

[295] Xue Qin, Simin Luan, John See, Cong Yang, and Zhijun Li. EmbodiedGovBench: A benchmark for governance, recovery, and upgrade safety in embodied agent systems. arXiv preprint arXiv:2604.11174, 2026.

[296] Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, and Xuelong Li. Spatialvla: Exploring spatial representations for visual-language-action model. arXiv preprint arXiv:2501.15830, 2025.

[297] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[298] Harshitha Rajaprakash, Aditeya Prajapati, Rong Xue, Abrar Anwar, and Jesse Thomason. SAFECAST: Robust failure detection for VLA policies with contrast-set training and calibration. arXiv preprint arXiv:2608.04246, 2026.

[299] Krishan Rana, Jesse Haviland, Sourav Garg, Jad Abou-Chakra, Ian Reid, and Niko Suenderhauf. Sayplan: Grounding large language models using 3d scene graphs for scalable robot task planning. *arXiv preprint arXiv:2307.06135*, 2023.

[300] Krishan Rana, Jesse Haviland, Sourav Garg, Jad Abou-Chakra, Ian Reid, and Niko Suenderhauf. Sayplan: Grounding large language models using 3d scene graphs for scalable robot task planning. arXiv preprint arXiv:2307.06135, 2023.

[301] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. In *International Conference on Learning Representations*, volume 2025, pages 28085–28128, 2025.

[302] Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. arXiv preprint arXiv:1910.01708, 2019.

[303] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

[304] Allen Z. Ren, Jaden Clark, Anushri Dixit, Masha Itkina, Anirudha Majumdar, and Dorsa Sadigh. Explore until confident: Efficient exploration for embodied question answering. arXiv preprint arXiv:2403.15941, 2024.

[305] Allen Z. Ren, Jaden Clark, Anushri Dixit, Masha Itkina, Anirudha Majumdar, and Dorsa Sadigh. Explore until confident: Efficient exploration for embodied question answering. In *Proceedings of Robotics: Science and Systems*, 2024.

[306] Allen Z. Ren, Anushri Dixit, A. A. Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila A. Takayama, Fei Xia, Jake Varley, Zhenjia Xu, Dorsa Sadigh, Andy Zeng, and Anirudha Majumdar. Robots that ask for help: Uncertainty alignment for large language model planners. In *Proceedings of the 7th Conference on Robot Learning*, 2023.

[307] Moritz Reuss, Ömer Erdinç Yağmurlu, Fabian Wenzel, and Rudolf Lioutikov. Multimodal Diffusion Transformer: Learning versatile behavior from multimodal goals. *arXiv preprint arXiv:2407.05996*, 2024.

[308] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[309] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9338–9346. IEEE, 2019.

[310] Lukas Schmid, Marcus Abate, Yun Chang, and Luca Carlone. Khronos: A unified approach for spatio-temporal metric-semantic slam in dynamic environments. *arXiv preprint arXiv:2402.13817*, 2024.

[311] Lukas Schmid, Jeffrey Delmerico, Johannes Schönberger, Juan Nieto, Marc Pollefeys, Roland Siegwart, and Cesar Cadena. Panoptic multi-tsdfs: a flexible representation for online multi-resolution volumetric mapping and long-term dynamic scene consistency. arXiv preprint arXiv:2109.10165, 2022.

[312] Tim Schneider, Boris Belousov, Georgia Chalvatzaki, Diego Romeres, Devesh K. Jha, and Jan Peters. Active exploration for robotic manipulation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[313] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5B: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022.

[314] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

[315] Junwon Seo, Kensuke Nakamura, and Andrea V. Bajcsy. Uncertainty-aware latent safety filters for avoiding out-of-distribution failures. arXiv preprint arXiv:2505.00779, 2025.

[316] Pierre Sermanet, Tianli Ding, Jeffrey Zhao, Fei Xia, Debidatta Dwibedi, Keerthana Gopalakrishnan, Christine Chan, Gabriel Dulac-Arnold, Sharath Maddineni, Nikhil J Joshi, et al. RoboVQA: Multimodal Long-Horizon Reasoning for Robotics. *arXiv preprint arXiv:2311.00899*, 2023.

[317] Pierre Sermanet, Anirudha Majumdar, Alex Irpan, Dmitry Kalashnikov, and Vikas Sindhwani. Generating robot constitutions and benchmarks for semantic safety. arXiv preprint arXiv:2503.08663, 2025.

[318] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems*, 36:38154–38180, 2023.

[319] Lucy Xiaoyang Shi, Zheyuan Hu, Tony Z. Zhao, Archit Sharma, Karl Pertsch, Jianlan Luo, Sergey Levine, and Chelsea Finn. Yell At Your Robot: Improving On-the-Fly from Language Corrections. *arXiv preprint arXiv:2403.12910*, 2024.

[320] Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, et al. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. *arXiv preprint arXiv:2502.19417*, 2025.

[321] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning. *arXiv preprint arXiv:2303.11366*, 2023.

[322] Mohit Shridhar and David Hsu. Interactive visual grounding of referring expressions for human-robot interaction. arXiv preprint arXiv:1806.03831, 2018.

[323] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. arXiv preprint arXiv:2209.05451, 2022.

[324] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 10737–10746. IEEE, 2020.

[325] Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. SmolVLA: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

[326] Bingrui Sima, Lizhong Wang, Xiaoya Lu, Kun He, and Xiao Yang. Self-evolving just-in-time memory for proactive embodied safety. Lecture Notes in Computer Science, 2026.

[327] Harsh Singh, Rocktim Jyoti Das, Mingfei Han, Preslav Nakov, and Ivan Laptev. MALMM: Multi-Agent Large Language Models for Zero-Shot Robotic Manipulation. *arXiv preprint arXiv:2411.17636*, 2024.

[328] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. *arXiv preprint arXiv:2209.11302*, 2022.

[329] Rohan Sinha, Anushri Dixit, Ran Tian, Anirudha Majumdar, and Andrea Bajcsy. Rethinking safety for generalist robots, 2026.

[330] David Snyder, Apurva Badithela, Nikolai Matni, George J. Pappas, Anirudha Majumdar, Masha Itkina, and Haruki Nishimura. Beyond binary success: Sample-efficient and statistically rigorous robot policy comparison. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026.

[331] Yejin Son, Minseo Kim, Sungwoong Kim, Seungju Han, Jian Kim, Dongju Jang, Youngjae Yu, and Chanyoung Park. Subtle risks, critical failures: A framework for diagnosing physical safety of llms for embodied decision making. Proceedings of EMNLP, 2025.

[332] 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. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15768–15780. IEEE, 2025.

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

[334] Xiangyu Su, Juzhan Xu, Oliver van Kaick, Kai Xu, and Ruizhen Hu. Imr-llm: Industrial multi-robot task planning and program generation using large language models. *arXiv preprint arXiv:2603.02669*, 2026.

[335] Peize Sun, Shoufa Chen, Chenchen Zhu, Fanyi Xiao, Ping Luo, Saining Xie, and Zhicheng Yan. Going denser with open-vocabulary part segmentation. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15407–15419. IEEE, 2023.

[336] Zhen Sun, Yongjian Guo, Haoran Sun, Luqiao Wang, Wei Lu, Jiachi Ji, Shengzhe Ji, Junwu Xiong, and Zhijun Meng. Pre-vla: Preemptive runtime verification for reliable vision-language-action and world-model rollouts. arXiv preprint arXiv:2605.22446, 2026.

[337] Sudharshan Suresh, Haozhi Qi, Tingfan Wu, Taosha Fan, Luis Pineda, Mike Lambeta, Jitendra Malik, Mrinal Kalakrishnan, Roberto Calandra, Michael Kaess, Joseph Ortiz, and Mustafa Mukadam. Neural feels with neural fields: Visuo-tactile perception for in-hand manipulation. arXiv preprint arXiv:2312.13469, 2023.

[338] Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. arXiv preprint arXiv:2303.08128, 2023.

[339] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in neural information processing systems*, 34:251–266, 2021.

[340] Ihab Tabbara, Yuxuan Yang, Ahmad Hamzeh, Maxwell Astafyev, and Hussein Sibai. Designing latent safety filters using pre-trained vision models. arXiv preprint arXiv:2509.14758, 2025.

[341] Hengkai Tan, Yao Feng, Xinyi Mao, Shuhe Huang, Guodong Liu, Zhongkai Hao, Hang Su, and Jun Zhu. AnyPos: Automated Task-Agnostic Actions for Bimanual Manipulation. *arXiv preprint arXiv:2507.12768*, 2025.

[342] Huajie Tan, Enshen Zhou, Zhiyu Li, Yijie Xu, Yuheng Ji, Xiansheng Chen, Cheng Chi, Pengwei Wang, Huizhu Jia, Yulong Ao, Mingyu Cao, Sixiang Chen, Zhe Li, Mengzhen Liu, Zixiao Wang, Shanyu Rong, Yaoxu Lyu, Zhongxia Zhao, Peterson Co, Yibo Li, Yi Han, Shaoxuan Xie, Guocai Yao, Songjing Wang, Leiduo Zhang, Xi Yang, Yance Jiao, Donghai Shi, Kunchang Xie, Shaokai Nie, Chunlei Men, Yonghua Lin, Zhongyuan Wang, Tiejun Huang, and Shanghang Zhang. Robobrain 2.5: Depth in sight, time in mind. arXiv preprint arXiv:2601.14352, 2026.

[343] Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-Kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Demonstrating gpu parallelized robot simulation and rendering for generalizable embodied ai with maniskill3. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, 2025.

[344] BAAI RoboBrain Team, Mingyu Cao, Huajie Tan, Yuheng Ji, Xiansheng Chen, Minglan Lin, Zhiyu Li, Zhou Cao, Pengwei Wang, Enshen Zhou, Yi Han, Yingbo Tang, Xiangqi Xu, Wei Guo, Yaoxu Lyu, Yijie Xu, Jiayu Shi, Mengfei Du, Cheng Chi, Mengdi Zhao, Xiaoshuai Hao, Junkai Zhao, Xiaojie Zhang, Shanyu Rong, Huaihai Lyu, Zhengliang Cai, Yankai Fu, Ning Chen, Bolun Zhang, Lingfeng Zhang, Shuyi Zhang, Dong Liu, Xi Feng, Songjing Wang, Xiaodan Liu, Yance Jiao, Mengsi Lyu, Zhuo Chen, Chenrui He, Yulong Ao, Xue Sun, Zheqi He, Jingshu Zheng, Xi Yang, Donghai Shi, Kunchang Xie, Bochao Zhang, Shaokai Nie, Chunlei Men, Yonghua Lin, Zhongyuan Wang, Tiejun Huang, and Shanghang Zhang. Robobrain 2.0 technical report. arXiv preprint arXiv:2507.02029, 2025.

[345] Gemini Robotics Team, Abbas Abdolmaleki, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Ashwin Balakrishna, Nathan Batchelor, Alex Bewley, Jeff Bingham, Michael Bloesch, Konstantinos Bousmalis, Philemon Brakel, Anthony Brohan, Thomas Buschmann, Arunkumar Byravan, Serkan Cabi, Ken Caluwaerts, Federico Casarini, Christine Chan, Oscar Chang, London Chappellet-Volpini, Jose Enrique Chen, Xi Chen, Hao-Tien Lewis Chiang, Krzysztof Choromanski, Adrian Collister, David B. D'Ambrosio, Sudeep Dasari, Todor Davchev, Meet Kirankumar Dave, Coline Devin, Norman Di Palo, Tianli Ding, Carl Doersch, Adil Dostmohamed, Yilun Du, Debidatta Dwibedi, Sathish Thoppay Egambaram, Michael Elabd, Tom Erez, Xiaolin Fang, Claudio Fantacci, Cody Fong, Erik Frey, Chuyuan Fu, Ruiqi Gao, Marissa Giustina, Keerthana Gopalakrishnan, Laura Graesser, Oliver Groth, Agrim Gupta, Roland Hafner, Steven Hansen, Leonard Hasenclever, Sam Haves, Nicolas Heess, Brandon Hernaez, Alex Hofer, Jasmine Hsu, Lu Huang, Sandy H. Huang, Atil Iscen, Mithun George Jacob, Deepali Jain, Sally Jesmonth, Abhishek Jindal, Ryan Julian, Dmitry Kalashnikov, M. Emre Karagozler, Stefani Karp, Matija Kecman, J. Chase Kew, Donnie Kim, Frank Kim, Junkyung Kim, Thomas Kipf, Sean Kirmani, Ksenia Konyushkova, Li Yang Ku, Yuheng Kuang, Thomas Lampe, Antoine Laurens, Tuan Anh Le, Isabel Leal, Alex X. Lee, Tsang-Wei Edward Lee, Guy Lever, Jacky Liang, Li-Heng Lin, Fangchen Liu, Shangbang Long, Caden Lu, Sharath Maddineni, Anirudha Majumdar, Kevis-Kokitsi Maninis, Andrew Marmon, Sergio Martinez, Assaf Hurwitz Michaely, Niko Milonopoulos, Joss Moore, Robert Moreno, Michael Neunert, Francesco Nori, Joy Ortiz, Kenneth Oslund, Carolina Parada, Emilio Parisotto, Amaris Paryag, Acorn Pooley, Thomas Power, Alessio Quaglino, Haroon Qureshi, Rajkumar Vasudeva Raju, Helen Ran, Dushyant Rao, Kanishka Rao, Isaac Reid, David Rendleman, Krista Reymann, Miguel Rivas, Francesco Romano, Yulia Rubanova, Peter Pastor Sampedro, Pannag R Sanketi, Dhruv Shah, Mohit Sharma, Kathryn Shea, Mohit Shridhar, Charles Shu, Vikas Sindhwani, Sumeet Singh, Radu Soricut, Rachel Sterneck, Ian Storz, Razvan Surdulescu, Jie Tan, Jonathan Tompson, Saran Tunyasuvunakool, Jake Varley, Grace Vesom, Giulia Vezzani, Maria Bauza Villalonga, Oriol Vinyals, René Wagner, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Chengda Wu, Markus Wulfmeier, Fei Xia, Ted Xiao, Annie Xie, Jinyu Xie, Peng Xu, Sichun Xu, Ying Xu, Zhuo Xu, Jimmy Yan, Sherry Yang, Skye Yang, Yuxiang Yang, Hiu Hong Yu, Wenhao Yu, Wentao Yuan, Yuan Yuan, Jingwei Zhang, Tingnan Zhang, Zhiyuan Zhang, Allan Zhou, Guangyao Zhou, and Yuxiang Zhou. Gemini robotics 1.5: Pushing the frontier of generalist robots with advanced embodied reasoning, thinking, and motion transfer. arXiv preprint arXiv:2510.03342, 2025.

[346] Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez

Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, Steven Bohez, Konstantinos Bousmalis, Anthony Brohan, Thomas Buschmann, Arunkumar Byravan, Serkan Cabi, Ken Caluwaerts, Federico Casarini, Oscar Chang, Jose Enrique Chen, Xi Chen, Hao-Tien Lewis Chiang, Krzysztof Choromanski, David D'Ambrosio, Sudeep Dasari, Todor Davchev, Coline Devin, Norman Di Palo, Tianli Ding, Adil Dostmohamed, Danny Driess, Yilun Du, Debidatta Dwibedi, Michael Elabd, Claudio Fantacci, Cody Fong, Erik Frey, Chuyuan Fu, Marissa Giustina, Keerthana Gopalakrishnan, Laura Graesser, Leonard Hasenclever, Nicolas Heess, Brandon Hernaez, Alexander Herzog, R. Alex Hofer, Jan Humplik, Atil Iscen, Mithun George Jacob, Deepali Jain, Ryan Julian, Dmitry Kalashnikov, M. Emre Karagozler, Stefani Karp, Chase Kew, Jerad Kirkland, Sean Kirmani, Yuheng Kuang, Thomas Lampe, Antoine Laurens, Isabel Leal, Alex X. Lee, Tsang-Wei Edward Lee, Jacky Liang, Yixin Lin, Sharath Maddineni, Anirudha Majumdar, Assaf Hurwitz Michaely, Robert Moreno, Michael Neunert, Francesco Nori, Carolina Parada, Emilio Parisotto, Peter Pastor, Acorn Pooley, Kanishka Rao, Krista Reymann, Dorsa Sadigh, Stefano Saliceti, Pannag Sanketi, Pierre Sermanet, Dhruv Shah, Mohit Sharma, Kathryn Shea, Charles Shu, Vikas Sindhwani, Sumeet Singh, Radu Soricut, Jost Tobias Springenberg, Rachel Sterneck, Razvan Surdulescu, Jie Tan, Jonathan Tompson, Vincent Vanhoucke, Jake Varley, Grace Vesom, Giulia Vezzani, Oriol Vinyals, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Fei Xia, Ted Xiao, Annie Xie, Jinyu Xie, Peng Xu, Sichun Xu, Ying Xu, Zhuo Xu, Yuxiang Yang, Rui Yao, Sergey Yaroshenko, Wenhao Yu, Wentao Yuan, Jingwei Zhang, Tingnan Zhang, Allan Zhou, and Yuxiang Zhou. Gemini robotics: Bringing ai into the physical world. arXiv preprint arXiv:2503.20020, 2025.

[347] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, March 2002.

[348] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[349] Mahiro Ukai, Shuhei Kurita, and Nakamasa Inoue. Status bench: A rigorous benchmark for evaluating object state understanding in vision-language models. arXiv preprint arXiv:2510.22571, 2025.

[350] Francisco J Varela, Evan Thompson, and Eleanor Rosch. *The embodied mind: Cognitive science and human experience*. The MIT press, 2017.

[351] Valeria Villani, Fabio Pini, Francesco Leali, and Cristian Secchi. Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications. *Mechatronics*, 55:248–266, 2018.

[352] Kim P. Wabersich and Melanie N. Zeilinger. A predictive safety filter for learning-based control of constrained nonlinear dynamical systems. *Automatica*, 129:109597, 2021.

[353] Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, et al. BridgeData V2: A Dataset for Robot Learning at Scale. *arXiv preprint arXiv:2308.12952*, 2023.

[354] Yang Wan, Xihang Yue, Zhirui Liu, Ziyuan Chu, Shuxun Wang, Yuhan Chen, Xiaonan Jiang, Xukun Zhu, Yubo Dong, and Linchao Zhu. No free checker: A survey of verifiers for robot policies. arXiv preprint arXiv:2609.09250, 2026.

[355] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An Open-Ended Embodied Agent with Large Language Models. *arXiv preprint arXiv:2305.16291*, 2023.

[356] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J. Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. arXiv preprint arXiv:1901.02970, 2019.

[357] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. arXiv preprint arXiv:2503.11651, 2025.

[358] Junjian Wang, Lidan Zhao, and Xi Sheryl Zhang. MADRA: Multi-Agent Debate for Risk-Aware Embodied Planning. *arXiv preprint arXiv:2511.21460*, 2025.

[359] Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pages 2609–2634, 2023.

[360] Lirui Wang, Yiyang Ling, Zhecheng Yuan, Mohit Shridhar, Chen Bao, Yuzhe Qin, Bailin Wang, Huazhe Xu, and Xiaolong Wang. GenSim: Generating Robotic Simulation Tasks via Large Language Models. In *International Conference on Learning Representations*, 2024.

[361] Maggie Wang, Stephen Tian, Aiden Swann, Ola Shorinwa, Jiajun Wu, and Mac Schwager. Phys2real: Fusing vlm priors with interactive online adaptation for uncertainty-aware sim-to-real manipulation. arXiv preprint arXiv:2510.11689, 2026.

[362] Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. arXiv preprint arXiv:2501.12387, 2025.

[363] Qineng Wang, Wenlong Huang, Yu Zhou, Hang Yin, Tianwei Bao, Jianwen Lyu, Weiyu Liu, Ruohan Zhang, Jiajun Wu, Li Fei-Fei, and Manling Li. Enact: Evaluating embodied cognition with world modeling of egocentric interaction. In *International Conference on Learning Representations*, 2026.

[364] Qineng Wang, Baiqiao Yin, Pingyue Zhang, Jianshu Zhang, Kangrui Wang, Zihan Wang, Jieyu Zhang, Keshigeyan Chandrasegaran, Han Liu, Ranjay Krishna, Saining Xie, Jiajun Wu, Li Fei-Fei, and Manling Li. Mindcube: Spatial mental modeling from limited views. arXiv preprint arXiv:2506.21458, 2026.

[365] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. arXiv preprint arXiv:2312.14132, 2024.

[366] Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, et al. OpenHands: An Open Platform for AI Software Developers as Generalist Agents. In *International Conference on Learning Representations*, 2025.

[367] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *International Conference on Learning Representations*, 2023.

[368] Yixuan Wang, Rhythm Syed, Fangyu Wu, Mengchao Zhang, Aykut Onol, Jose Barreiros, Hooshang Nayyeri, Tony Dear, Huan Zhang, and Yunzhu Li. Interactive world simulator for robot policy training and evaluation. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026.

[369] Yufei Wang, Zhanyi Sun, Jesse Zhang, Zhou Xian, Erdem Biyik, David Held, and Zackory Erickson. RL-VLM-F: Reinforcement learning from vision language foundation model feedback. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

[370] Yufei Wang, Zhou Xian, Feng Chen, Tsun-Hsuan Wang, Yian Wang, Katerina Fragkiadaki, Zackory Erickson, David Held, and Chuang Gan. RoboGen: Towards Unleashing Infinite Data for Automated Robot Learning via Generative Simulation. In *International Conference on Machine Learning*, 2024.

[371] Yuran Wang, Siqiao Huang, Mingleyang Li, Chenhao Zhang, Jiaqi Liang, Weiyang Jin, Yue Chen, Xuemin Chi, Donghao Zhou, Qize Yu, et al. Openwam: An open, modular exploration towards systematic world-action model pretraining. *arXiv preprint arXiv:2609.07398*, 2026.

[372] Zhenyu Wei, Zhixuan Xu, Jingxiang Guo, Yiwen Hou, Chongkai Gao, Zhehao Cai, Jiayu Luo, and Lin Shao. D(R,O) grasp: A unified representation of robot and object interaction for cross-embodiment dexterous grasping. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4982–4988. IEEE, 2025.

[373] Bowen Wen, Jonathan Tremblay, Valts Blukis, Stephen Tyree, Thomas Muller, Alex Evans, Dieter Fox, Jan Kautz, and Stan Birchfield. Bundlesdf: Neural 6-dof tracking and 3d reconstruction of unknown objects. arXiv preprint arXiv:2303.14158, 2023.

[374] Bowen Wen, Matthew Trepte, Joseph Aribido, Jan Kautz, Orazio Gallo, and Stan Birchfield. Foundationstereo: Zero-shot stereo matching. arXiv preprint arXiv:2501.09898, 2025.

[375] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. arXiv preprint arXiv:2312.08344, 2024.

[376] Junjie Wen, Yichen Zhu, Jinming Li, Zhibin Tang, Chaomin Shen, and Feifei Feng. DexVLA: Vision-language model with plug-in diffusion expert for general robot control. In *Proceedings of the 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 3094–3114. PMLR, 2025.

[377] Diankun Wu, Fangfu Liu, Yi-Hsin Hung, and Yueqi Duan. Spatial-mllm: Boosting mllm capabilities in visual-based spatial intelligence. *Advances in neural information processing systems*, 38:13569–13597, 2026.

[378] Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. In *International Conference on Learning Representations (ICLR)*, 2024.

[379] Jiajun Wu, Ilker Yildirim, Joseph J. Lim, William T. Freeman, and Joshua B. Tenenbaum. Galileo: Perceiving Physical Object Properties by Integrating a Physics Engine with Deep Learning. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

[380] Kun Wu, Chengkai Hou, Jiaming Liu, Zhengping Che, Xiaozhu Ju, Zhuqin Yang, Meng Li, Yinuo Zhao, Zhiyuan Xu, Guang Yang, et al. RoboMIND: Benchmark on Multi-embodiment Intelligence Normative Data for Robot Manipulation. *arXiv preprint arXiv:2412.13877*, 2024.

[381] Tao Wu, Chuhao Zhou, Guangyu Zhao, Haozhi Cao, Yewen Pu, and Jianfei Yang. When robots should say "i don't know": Benchmarking abstention in embodied question answering. arXiv preprint arXiv:2512.04597, 2025.

[382] Yike Wu, Jiatao Zhang, Nan Hu, Lanling Tang, Guilin Qi, Jun Shao, Jie Ren, and Wei Song. Mldt: Multi-level decomposition for complex long-horizon robotic task planning with open-source large language model. In *International Conference on Database Systems for Advanced Applications*, pages 251–267. Springer, 2024.

[383] Zhiyuan Wu, Rolandos Alexandros Potamias, Xuyang Zhang, Zhongqun Zhang, Jiankang Deng, and Shan Luo. CEDex: Cross-embodiment dexterous grasp generation at scale from human-like contact representations. In *2026 IEEE International Conference on Robotics and Automation (ICRA)*, 2026.

[384] Chong Xia, Kai Zhu, Zizhuo Wang, Fangfu Liu, Zhizheng Zhang, and Yueqi Duan. SimRecon: SimReady Compositional Scene Reconstruction from Real Videos. *arXiv preprint arXiv:2603.02133*, 2026.

[385] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. arXiv preprint arXiv:1711.00199, 2018.

[386] Yuxi Xiao, Longfei Li, Shen Yan, Xinhang Liu, Sida Peng, Yunchao Wei, Xiaowei Zhou, and Bingyi Kang. Spatialtree: How spatial abilities branch out in mllms. arXiv preprint arXiv:2512.20617, 2026.

[387] Quanting Xie, So Yeon Min, Pengliang Ji, Yue Yang, Tianyi Zhang, Kedi Xu, Aarav Bajaj, Ruslan Salakhutdinov, Matthew Johnson-Roberson, and Yonatan Bisk. Embodied-rag: General non-parametric embodied memory for retrieval and generation. arXiv preprint arXiv:2409.18313, 2025.

[388] Junwu Xiong, Jiaxuan Gao, Wei Chai, Renxing Chen, Yuzhen Li, Yu Guo, Yucheng Guo, Mingxi Luo, Wenyang Ma, Yiyun Mou, Yifei Zhang, Chen Zhou, et al. Embodied operators and benchmarking: Toward reusable and deployable embodied intelligence systems. arXiv preprint arXiv:2607.03283, 2026.

[389] Mengda Xu, Han Zhang, Yifan Hou, Zhenjia Xu, Linxi Fan, Manuela Veloso, and Shuran Song. DexUMI: Using human hand as the universal manipulation interface for dexterous manipulation. In *Proceedings of the 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 437–459. PMLR, 2025.

[390] Xiaomeng Xu, Yifan Hou, Chendong Xin, Zeyi Liu, and Shuran Song. Compliant Residual DAgger: Improving Real-World Contact-Rich Manipulation with Human Corrections. *arXiv preprint arXiv:2506.16685*, 2025.

[391] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation'*, CIRA-97, page 146–151. IEEE Comput. Soc. Press, 1997.

[392] Bowen Yan, Jiahao Xiao, Kehui Liu, Jianbo Zhang, Zicheng Zhang, Qi Jia, Zhongjie Jia, Haoming Song, Chunyi Li, Bin Zhao, and Guangtao Zhai. Can vlms diagnose and recover from vla manipulation faults? In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 143967–143983. PMLR, 2026.

[393] Zhijie Yan, Shufei Li, Zuoxu Wang, Lixiu Wu, Han Wang, Jun Zhu, Lijiang Chen, and Jihong Liu. Dynamic open-vocabulary 3d scene graphs for long-term language-guided mobile manipulation. *IEEE Robotics and Automation Letters*, 10(5):4252–4259, 2025.

[394] Huaigang Yang, Ya Li, Min Ren, Bo Dai, Zhenliang Zhang, and Zhaofeng He. Saferelbench: A spatial-relation-aware benchmark for process-level safety in vlm-driven embodied agents. arXiv preprint arXiv:2607.14543, 2026.

[395] Jianwei Yang, Reuben Tan, Qianhui Wu, Ruijie Zheng, Baolin Peng, Yongyuan Liang, Yu Gu, Mu Cai, Seonghyeon Ye, Joel Jang, Yuquan Deng, Lars Liden, and Jianfeng Gao. Magma: A foundation model for multimodal ai agents. arXiv preprint arXiv:2502.13130, 2025.

[396] Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. arXiv preprint arXiv:2310.11441, 2023.

[397] 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. arXiv preprint arXiv:2412.14171, 2025.

[398] John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering. *arXiv preprint arXiv:2405.15793*, 2024.

[399] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. arXiv preprint arXiv:2401.10891, 2024.

[400] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. arXiv preprint arXiv:2406.09414, 2024.

[401] Rui Yang, Hanyang Chen, Junyu Zhang, Mark Zhao, Cheng Qian, Kangrui Wang, Qineng Wang, Teja Venkat Koripella, Marziyeh Movahedi, Manling Li, Heng Ji, Huan Zhang, and Tong Zhang. Embodiedbench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 70576–70631. PMLR, 2025.

[402] Sihan Yang, Runsen Xu, Yiman Xie, Sizhe Yang, Mo Li, Jingli Lin, Chenming Zhu, Xiaochen Chen, Haodong Duan, Xiangyu Yue, Dahua Lin, Tai Wang, and Jiangmiao Pang. Mmsi-bench: A benchmark for multi-image spatial intelligence. arXiv preprint arXiv:2505.23764, 2026.

[403] Sizhe Yang, Linning Xu, Hao Li, Juncheng Mu, Jia Zeng, Dahua Lin, and Jiangmiao Pang. Robo3r: Enhancing robotic manipulation with accurate feed-forward 3d reconstruction. arXiv preprint arXiv:2602.10101, 2026.

[404] Xuning Yang, Rishit Dagli, Alex Zook, Hugo Hadfield, Ankit Goyal, Stan Birchfield, Fabio Ramos, and Jonathan Tremblay. Robolab: A high-fidelity simulation benchmark for analysis of task generalist policies. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026.

[405] Yuncong Yang, Han Yang, Jiachen Zhou, Peihao Chen, Hongxin Zhang, Yilun Du, and Chuang Gan. 3d-mem: 3d scene memory for embodied exploration and reasoning. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17294–17303. IEEE, 2025.

[406] Ziyi Yang, Shreyas Sundara Raman, Ankit Shah, and Stefanie Tellex. Plug in the safety chip: Enforcing constraints for llm-driven robot agents. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[407] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

[408] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations*, 2023.

[409] 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.

[410] Zonghao Ying, Le Wang, Yisong Xiao, Jiakai Wang, Yingjun Ma, Jinyang Guo, Zhenfei Yin, Mingchuan Zhang, Aishan Liu, and Xianglong Liu. Agentsafe: Benchmarking the safety of embodied agents on hazardous instructions. arXiv preprint arXiv:2506.14697, 2025.

[411] Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. arXiv preprint arXiv:2312.03275, 2023.

[412] Tomoya Yoshida, Shuhei Kurita, Taichi Nishimura, and Shinsuke Mori. Text-driven affordance learning from egocentric vision. *arXiv preprint arXiv:2404.02523*, 2024.

[413] Jiawen Yu, Hairuo Liu, Qiaojun Yu, Jieji Ren, Ce Hao, Haitong Ding, Guangyu Huang, Guofan Huang, Yan Song, Panpan Cai, Cewu Lu, and Wenqiang Zhang. Forcevla: Enhancing vla models with a force-aware moe for contact-rich manipulation. arXiv preprint arXiv:2505.22159, 2025.

[414] Samson Yu, Kelvin Lin, Anxing Xiao, Jiafei Duan, and Harold Soh. Octopi: Object property reasoning with large tactile-language models. arXiv preprint arXiv:2405.02794, 2024.

[415] Zecheng Yu, Yifei Huang, Ryosuke Furuta, Takuma Yagi, Yusuke Goutsu, and Yoichi Sato. Fine-grained affordance annotation for egocentric hand-object interaction videos. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023.

[416] Haoqi Yuan, Bohan Zhou, Yuhui Fu, and Zongqing Lu. Cross-embodiment dexterous grasping with reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2025.

[417] Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024.

[418] Wenzhen Yuan, Siyuan Dong, and Edward Adelson. Gelsight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12):2762, November 2017.

[419] Yifu Yuan, Haiqin Cui, Yaoting Huang, Yibin Chen, Fei Ni, Zibin Dong, Pengyi Li, Yan Zheng, Hongyao Tang, and Jianye Hao. Embodied-r1: Reinforced embodied reasoning for general robotic manipulation. arXiv preprint arXiv:2508.13998, 2026.

[420] Mert Yuksekgonul, Federico Bianchi, Pratyusha Kalluri, Dan Jurafsky, and James Zou. When and why vision-language models behave like bags-of-words, and what to do about it? *arXiv preprint arXiv:2210.01936*, 2022.

[421] Kevin Zakka, Baruch Tabanpour, Qiayuan Liao, Mustafa Haiderbhai, Samuel Holt, Jing Yuan Luo, Arthur Allshire, Erik Frey, Koushil Sreenath, Lueder Alexander Kahrs, Carmelo Sferrazza, Yuval Tassa, and Pieter Abbeel. Demonstrating mujoco playground. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, 2025.

[422] Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic Control via Embodied Chain-of-Thought Reasoning. *arXiv preprint arXiv:2407.08693*, 2024.

[423] Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. arXiv preprint arXiv:2407.08693, 2025.

[424] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3D Diffusion Policy: Generalizable visuomotor policy learning via simple 3d representations. In *Robotics: Science and Systems (RSS)*, 2024.

[425] Andy Zeng, Maria Attarian, Brian Ichter, Krzysztof Choromanski, Adrian Wong, Stefan Welker, Federico Tombari, Aveek Purohit, Michael Ryoo, Vikas Sindhwani, Johnny Lee, Vincent Vanhoucke, and Pete Florence. Socratic models: Composing zero-shot multimodal reasoning with language. arXiv preprint arXiv:2204.00598, 2022.

[426] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, pages 726–747. PMLR, 2021.

[427] Haishan Zeng, Mengna Wang, and Peng Li. Emboteam: Grounding llm reasoning into reactive behavior trees via pddl for embodied multi-robot collaboration. *arXiv preprint arXiv:2601.11063*, 2026.

[428] Lihan Zha, Yuchen Cui, Li-Heng Lin, Minae Kwon, Montserrat Gonzalez Arenas, Andy Zeng, Fei Xia, and Dorsa Sadigh. Distilling and Retrieving Generalizable Knowledge for Robot Manipulation via Language Corrections. *arXiv preprint arXiv:2311.10678*, 2023.

[429] Borong Zhang, Yuhao Zhang, Jiaming Ji, Yingshan Lei, Yishuai Cai, Juntao Dai, Yuanpei Chen, and Yaodong Yang. Safevla: Towards safety alignment of vision-language-action model via constrained learning. arXiv preprint arXiv:2503.03480, 2025.

[430] Chuye Zhang, Yifei Simon Shao, Harshil Parekh, Junyao Shi, Pratik Chaudhari, Vijay Kumar, and Nadia Figueroa. Don't Yell at Your Robot: Physical Correction as the Collaborative Interface for Language Model Powered Robots. *arXiv preprint arXiv:2412.12602*, 2024.

[431] Hanbo Zhang, Yunfan Lu, Cunjun Yu, David Hsu, Xuguang Lan, and Nanning Zheng. Invigorate: Interactive visual grounding and grasping in clutter. arXiv preprint arXiv:2108.11092, 2024.

[432] Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin Godel Machine: Open-Ended Evolution of Self-Improving Agents. *arXiv preprint arXiv:2505.22954*, 2025.

[433] Kaifeng Zhang, Baoyu Li, Kris Hauser, and Yunzhu Li. Adaptigraph: Material-adaptive graph-based neural dynamics for robotic manipulation. arXiv preprint arXiv:2407.07889, 2024.

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

[435] Wenbo Zhang, Kaixuan Wang, Yutao Ouyang, Xiaoyu Huang, Liyang Li, Kailun Su, Weiyang Jin, Wenhao Chai, Haotian Liang, Zhiyang Dou, et al. An unexpected robot policy: Early evaluations of gpt-6 astra on robodojo and beyond. *arXiv preprint arXiv:2609.24170*, 2026.

[436] Yang Zhang, Shixin Yang, Chenjia Bai, Fei Wu, Xiu Li, Zhen Wang, and Xuelong Li. Towards Efficient LLM Grounding for Embodied Multi-Agent Collaboration. *arXiv preprint arXiv:2405.14314*, 2024.

[437] Yixian Zhang, Huanming Zhang, Feng Gao, Xiao Li, Zhihao Liu, Chunyang Zhu, Jiaxing Qiu, Yuchen Yan, Jiyuan Liu, Wenhao Tang, et al. Harness vla: Steering frozen vlas into reliable manipulation primitives via memory-guided agents. *arXiv preprint arXiv:2607.08448*, 2026.

[438] Yixian Zhang, Huanming Zhang, Feng Gao, Xiao Li, Zhihao Liu, Chunyang Zhu, Jiaxing Qiu, Yuchen Yan, Jiyuan Liu, Wenhao Tang, et al. Harness vla: Steering frozen vlas into reliable manipulation primitives via memory-guided agents. *arXiv preprint arXiv:2607.08448*, 2026.

[439] Zheyuan Zhang, Fengyuan Hu, Jayjun Lee, Freda Shi, Parisa Kordjamshidi, Joyce Chai, and Ziqiao Ma. Do vision-language models represent space and how? evaluating spatial frame of reference under ambiguities. arXiv preprint arXiv:2410.17385, 2025.

[440] Zongzheng Zhang, Haobo Xu, Zhuo Yang, Chenghao Yue, Zehao Lin, Huan ang Gao, Ziwei Wang, and Hao Zhao. Ta-vla: Elucidating the design space of torque-aware vision-language-action models. arXiv preprint arXiv:2509.07962, 2025.

[441] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, Ankur Handa, Tsung-Yi Lin, Gordon Wetzstein, Ming-Yu Liu, and Donglai Xiang. CoT-VLA: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1702–1713, 2025.

[442] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

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

[444] Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In *International Conference on Learning Representations (ICLR)*, 2026.

[445] Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuzhan Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, and Joseph E. Gonzalez. SGLang: Efficient controlled generation with RadixAttention. In *NeurIPS*, 2024.

[446] Yandan Zheng, Haoran Luo, Zhenghong Lin, Wenjin Liu, and Luu Anh Tuan. BenchBench: Benchmarking automated benchmark generation. arXiv preprint arXiv:2603.20807, 2026.

[447] Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning acting and planning in language models. *arXiv preprint arXiv:2310.04406*, 2023.

[448] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.

[449] Enshen Zhou, Qi Su, Cheng Chi, Zhizheng Zhang, Zhongyuan Wang, Tiejun Huang, Lu Sheng, and He Wang. Code-as-monitor: Constraint-aware visual programming for reactive and proactive robotic failure detection. arXiv preprint arXiv:2412.04455, 2025.

[450] Siyuan Zhou, Yilun Du, Yuncong Yang, Lei Han, Peihao Chen, Dit-Yan Yeung, and Chuang Gan. Learning 3d persistent embodied world models. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[451] Zhiyuan Zhou, Pranav Atreya, You Liang Tan, Karl Pertsch, and Sergey Levine. Autoeval: Autonomous evaluation of generalist robot manipulation policies in the real world. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 1997–2017. PMLR, 2025.

[452] Zibo Zhou, Yue Hu, Lingkai Zhang, Zonglin Li, and Siheng Chen. Beliefmapnav: 3d voxel-based belief map for zero-shot object navigation. arXiv preprint arXiv:2506.06487, 2025.

[453] Xingyu Zhu, Wenshuo Han, Zhouyu Wang, Yuran Wang, Ruihai Wu, Hao Dong, Fan Tang, Hechang Chen, Hyung Jin Chang, and Yixing Gao. Flatlab: A unified methodology framework and simulation-based benchmark for robotic manipulation of flat objects. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 167064–167086. PMLR, 2026.