Visual Agents in the Era of Self-Evolution: Toward Recursive Self-Improvement

Dong Chen, Shuai Ge, Yu Fu, Xiangyu Ren, Zhihao Zhang, Hao Wang, Changshuo Wang, Jiayi Ji, Henghui Ding, and Shuting He

***Abstract*—Large language model (LLM) agents have emerged as a paradigm for goal-directed problem solving, leveraging reasoning, planning, and tool use to accomplish complex tasks. Beyond text-based applications, these capabilities are increasingly being extended to visual domains, where visual agents ground their decisions and actions in images, videos, and interactive feedback. This paper provides a systematic overview of recent advances in these visual agents. We first conceptualize visual agents as systems in which one or more models operate within an associated agent harness. We then categorize mechanisms for improving agent capabilities by the depth of experience-driven change in these systems, distinguishing within-task adaptation, self-evolution, and recursive self-improvement (RSI). For each level, we review representative methods across visual understanding, generation, computer use, and embodied interaction, examining their sources of experience, targets of change, and underlying mechanisms. Relevant benchmarks and evaluation protocols are also summarized, providing a reference for evaluating visual agents across diverse tasks and settings. To support further research, resources are available at http://visual-agent-survey.github.io/.**

***Index Terms*—Visual Agents, Within-Task Adaptation, Self-Evolution, Recursive Self-Improvement**

I. Introduction

In an emerging problem-solving paradigm, large language model (LLM) agents pursue goals by combining reasoning, planning, and tool use [1], [2]. They extend the capabilities of LLMs beyond text generation by interacting with external environments [3], [4]. Observations and feedback from these interactions guide subsequent decisions, allowing agents to gather information, check intermediate results, and revise their plans. Agents can also use experience from earlier attempts to improve their ability to solve tasks [5]–[7]. These capabilities support applications in information seeking [8], software development [9], and mathematical problem solving [10].

Visual agents extend this paradigm to visual perception and interaction tasks, including visual understanding [11]–[13], image and video generation and editing [14]–[16], GUI perception and grounding [17], computer use in web and desktop environments [18]–[20], and embodied interaction [21]–[23]. These tasks require agents to extract task-relevant information from visual inputs and reason about spatial and temporal relationships. Compared with agents relying mainly on textual observations, visual agents ground their decisions and actions in images, videos, and visual feedback, enabling interaction with visually changing environments [24], [25]. This integration enables agents to retrieve relevant visual evidence, inspect generated outputs, and select actions based on the observed environment state. Building on these capabilities, recent work explores how output and action feedback can support self-evolution and recursive self-improvement [26], [27].

The rapid development of visual agents and their expanding range of applications have motivated a growing body of work reviewing recent progress. While existing reviews of multimodal agents cover architectures, capabilities, and applications [24], [25], they do not focus on self-evolution or recursive self-improvement (RSI). In contrast, existing work on self-evolution and RSI primarily targets text-based or general-purpose agents, treating visual agents as at most one component among many [28]–[32]. Meanwhile, domain-specific reviews examine visual agents in areas such as visual generation, computer use, and embodied interaction [33]–[40], but each addresses only a narrow subset of the field rather than providing a systematic classification of how visual agents as a whole adapt and improve. These gaps motivate a dedicated and unified account of the mechanisms through which visual agents adapt and improve.

In this work, we view visual agents as systems comprising one or more models operating within an agent harness. Building on this view, we distinguish within-task adaptation, self-evolution, and recursive self-improvement (RSI) by the target, persistence, and subsequent use of experience-driven changes. We use this taxonomy to synthesize research on visual understanding, generation and editing, computer use, and embodied interaction. We also review benchmarks and evaluation protocols, discuss open challenges, and outline directions for future research.

Contributions. This paper provides a structured overview of research on visual agent adaptation and evolution. We first clarify the complementary roles of model capabilities and agent harnesses to establish a common conceptual foundation. Building on this foundation, we organize the main body of the paper around a taxonomy that distinguishes within-task adaptation, self-evolution, and RSI. For each category, we discuss representative methods across different visual domains. We further examine existing benchmarks and evaluation protocols to clarify the extent to which their results support claims of improvement. Finally, based on this analysis, we identify open challenges and outline promising directions for future research

D. Chen, S. Ge, Y. Fu, X. Ren, Z. Zhang, and S. He are with Shanghai University of Finance and Economics, China. (e-mail: shuting.he@sufe.edu.cn).

H. Wang is with Mohamed bin Zayed University of Artificial Intelligence, United Arab Emirates.

C. Wang is with University College London, United Kingdom.

J. Ji is with Xiamen University, China.

H. Ding is with Fudan University, China.

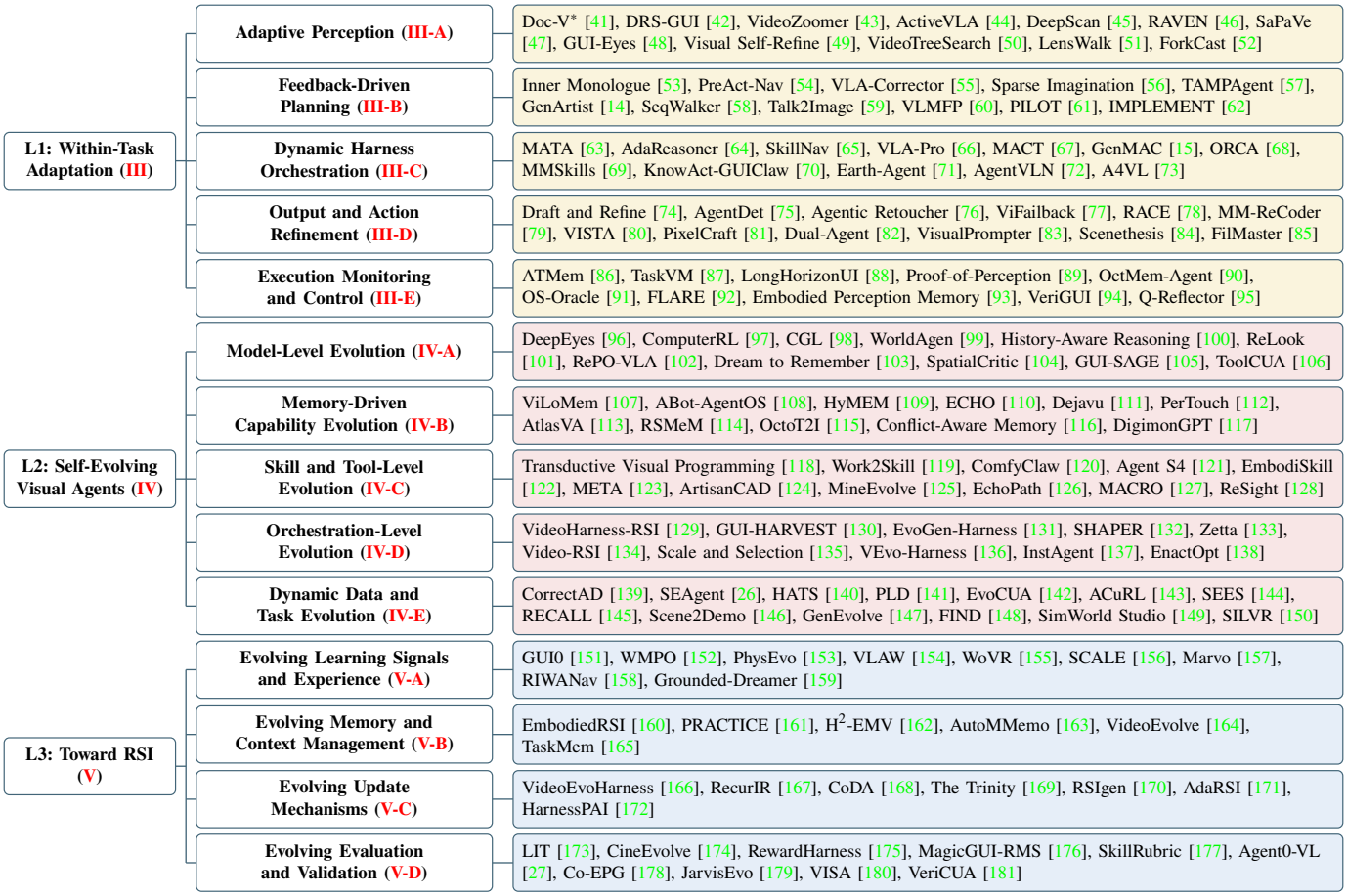


Fig. 1. **Taxonomy of visual agent adaptation and evolution.** The three levels cover within-task adaptation (L1), self-evolution (L2), and work toward recursive self-improvement (L3), as discussed in Sections III–V. Each branch identifies a mechanism subcategory, gives its subsection number, and lists representative works.

on visual agent adaptation and evolution.

Organization. Fig. 1 presents an overview of the taxonomy that structures this paper. Section II introduces the background of visual agents, including their definition and scope. It also discusses the complementary roles of models and agent harnesses and establishes the taxonomy of adaptation and evolution. Sections III, IV, and V examine existing research on within-task adaptation, self-evolution, and emerging approaches to recursive self-improvement, respectively. Section VI examines benchmarks and evaluation protocols for assessing these mechanisms. Section VII discusses open challenges and outlines promising directions for future research. Finally, Section VIII concludes the paper by summarizing its main insights.

II. Background

A. Definition and Scope

Definition. In this survey, a *visual agent* is a goal-directed system that uses visual observations to guide the selection and execution of task-level operations. These operations may include acquiring evidence, invoking tools, generating or editing visual content, and interacting with digital or physical environments. Its outputs may include responses grounded in visual evidence, visual artifacts, or actions in an environment.

Scope. We cover visual agents for visual understanding and for generating and editing visual content. Our review also extends to emerging interactive applications, including embodied interaction and computer use ranging from graphical user interface (GUI) perception and grounding to end-to-end workflows. We examine how an agent's own experience and external inputs such as human demonstrations and evaluation feedback can inform changes to task execution, agent resources, and improvement mechanisms.

B. Foundations of Visual Agents

A visual agent combines one or more models with an agent harness. Models provide internal capabilities for visual perception, planning, and output. The harness integrates orchestration, memory, reusable skills, tools, and monitoring and recovery mechanisms to coordinate execution and connect the agent to its environment [24], [25]. Training can improve model capabilities [182]–[184] and optimize harness policies for functions such as task scheduling and skill library maintenance [161], [185]. We write the agent state at evolution stage k as

$$\mathcal{A}_k = \big(\underbrace{\mathcal{M}_{\theta_k}}_{\text{models}}, \underbrace{\mathcal{H}_{\eta_k}}_{\text{harness}}\big), \tag{1}$$

where θ_k denotes the learned model parameters and η_k represents the configuration and resources of the harness. This formulation distinguishes updates to learned model capabilities

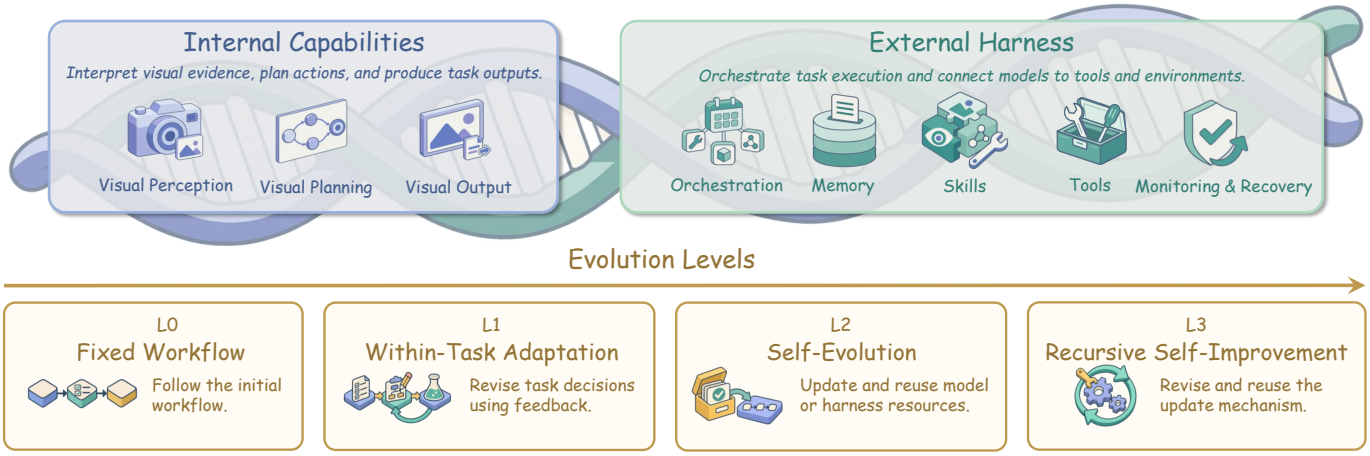


Fig. 2. **Visual agent foundations and levels of adaptation and evolution.** Models provide visual capabilities, and the harness orchestrates execution. The lower row shows a fixed workflow (L0), within-task adaptation from feedback (L1), self-evolution through persistent resource updates by a fixed updater (L2), and recursive self-improvement through reuse of a revised updater (L3).

from those to the external resources and procedures that support their use. Fig. 2 illustrates these complementary parts.

At execution step t, the agent combines its model component with its harness to act on the current task state. We formalize this process as

$$(a_t, z_{t+1}) = \mathcal{H}_{\eta_k}[\mathcal{M}_{\theta_k}](g, o_t, z_t, W_t). \tag{2}$$

Here, g denotes the task goal, and o_t represents the current visual observation. The execution strategy W_t is the active workflow for the current task. It specifies the selection and order of steps for evidence acquisition, planning, tool use, and control. The transient working state z_t records workflow progress and plan details for execution, such as grounded targets and operation arguments. The outputs a_t and z_{t+1} denote the operation executed at step t and the resulting working state, respectively. The operator $\mathcal{H}_{\eta_k}[\mathcal{M}_{\theta_k}]$ represents the agent's execution process as implemented by the model component and harness. The harness supplies context to model calls, invokes tools as needed, executes the selected operations, and incorporates outcome feedback into subsequent steps.

Before execution, the initial workflow W_0 is selected or instantiated based on the task goal and initial observations. We use fixed-workflow execution as the Level 0 (L0) reference, illustrated in Fig. 3(a). In this setting, the agent follows its initial workflow throughout the task [11], [12], so the active workflow at each execution step t satisfies

$$W_t = W_0. \tag{3}$$

Within this fixed workflow, the current visual observation o_t supplies input to the scheduled operation, and the working state z_t records execution progress. At the task-level, execution feedback does not revise current decisions or proposed outputs.

C. Taxonomy

Building on the L0 reference, our taxonomy defines Level 1 (L1) as within-task adaptation, Level 2 (L2) as self-evolution, and Level 3 (L3) as recursive self-improvement (RSI).

The index t tracks execution steps within a task, and k tracks stages of agent evolution. In Fig. 3(a) and (b), $\mathcal{A}_0$ denotes the

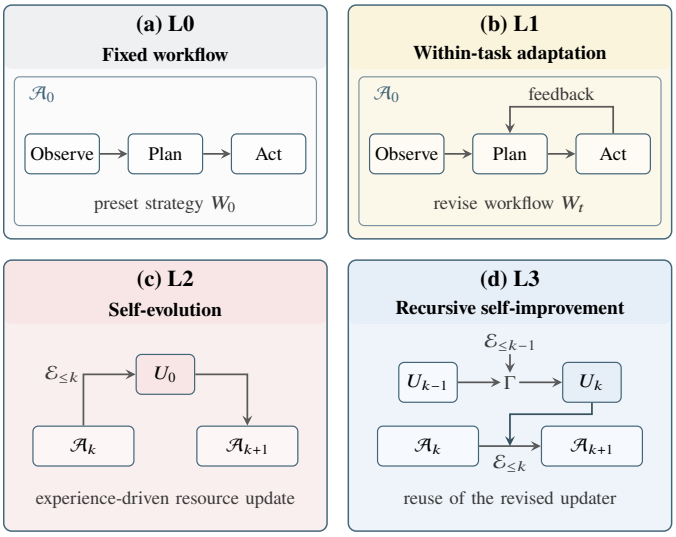


Fig. 3. **Fixed-workflow reference and L1–L3 processes.** (a) At L0, $\mathcal{A}_0$ follows the workflow W_0 set before task execution, so $W_t = W_0$ at every step t. (b) At L1, $\mathcal{A}_0$ can revise the active workflow W_t into W_{t+1} in response to task feedback. (c) At L2, the fixed updater U_0 uses experience $\mathcal{E}_{\le k}$ to update model or harness resources used in workflow execution, changing the agent state from $\mathcal{A}_k$ to $\mathcal{A}_{k+1}$. (d) At L3, Γ uses $\mathcal{E}_{\le k-1}$ to refine U_{k-1} into U_k. The revised updater then uses $\mathcal{E}_{\le k}$ to update the agent state from $\mathcal{A}_k$ to $\mathcal{A}_{k+1}$.

models and harness used to execute the illustrated workflows. At L1, the agent can revise the active workflow W_t in response to task feedback. Experience guides updates to model or harness resources at L2 and revisions to update mechanisms at L3. We distinguish these processes by what changes, whether the change persists, and how it is used later.

At L1, the agent uses new evidence to adapt its execution of the current task. It may change the active workflow, as illustrated in Fig. 3(b), or revise or reselect a proposed output or action sequence. For changes to the selected workflow steps or their order, we write

$$W_{t+1} = \Phi(W_t, o_{t+1}, f_t), \tag{4}$$

where f_t is the feedback available after step t and Φ is the workflow revision function. A fixed Φ can produce different

workflows as observations and feedback change.

At L2, the agent updates a model or harness resource using experience from its own interactions or training attempts. It retains the updated resource and uses it in a later task, trial, or training round. Human demonstrations, annotations, and external evaluations can provide additional supervision. At stage k, the available experience $\mathcal{E}_{\leq k}$ includes current observations, outcomes, and feedback, along with records retrieved from earlier stages. The fixed updater U_0 shown in Fig. 3(c) uses the agent state $\mathcal{A}_k$ from Eq. (1) and this experience to produce the next state as

$$\mathcal{A}_{k+1} = U_0(\mathcal{A}_k, \mathcal{E}_{\leq k}). \tag{5}$$

The updater can use this experience to propose, apply, and evaluate changes to the model or harness. In subsequent execution, the agent uses the models and harness in $\mathcal{A}_{k+1}$ to carry out the active workflow W_t. Several methods described as RSI fall under L2 because they reuse revised harnesses or skill libraries in later stages with a fixed updater [129], [134], [186].

Fig. 3(d) shows how the revision procedure Γ refines U_{k-1} into U_k using experience available at stage $k-1$. At stage k, the agent reuses U_k to update the state as

$$\mathcal{A}_{k+1} = U_k(\mathcal{A}_k, \mathcal{E}_{\leq k}). \tag{6}$$

At stage k, the agent also applies Γ to U_k to obtain the updater for stage $k+1$. A separate mechanism can carry out this step, or the updater can help modify itself. A fixed Γ_0 yields U_{k+1}. When the procedure also updates its own implementation, the current version Γ_k yields both U_{k+1} and the new procedure Γ_{k+1}. We write the two cases as

$$U_{k+1} = \Gamma_0(U_k, \mathcal{A}_k, \mathcal{E}_{\leq k}), \tag{7}$$

$$(U_{k+1}, \Gamma_{k+1}) = \Gamma_k(U_k, \mathcal{A}_k, \mathcal{E}_{\leq k}). \tag{8}$$

At stage $k+1$, U_{k+1} updates the state from Eq. (6) using the available experience as follows

$$\mathcal{A}_{k+2} = U_{k+1}(\mathcal{A}_{k+1}, \mathcal{E}_{\leq k+1}). \tag{9}$$

With either a fixed Γ_0 or an evolving Γ_k, this update meets the L3 criterion when it uses a modified component of U_{k+1}, such as a retrained evaluator whose scores on trial runs guide the policy update. In the second case, applying Γ_{k+1} to a later updater extends the recursion to the procedure itself.

III. L1 Within-Task Adaptation of Visual Agents

As introduced in Section II-C, within-task adaptation (L1) describes how a *visual agent* responds to new observations and action outcomes during a task. When the evidence leaves the next decision uncertain, the agent may need a better view before deciding how to proceed. A revised plan can in turn change which capabilities the harness invokes, and feedback during execution can reveal an output or action that needs correction. By monitoring the effects of its actions, the agent can judge whether to proceed, recover, or stop. Table I summarizes how these mechanisms act on different parts of execution.

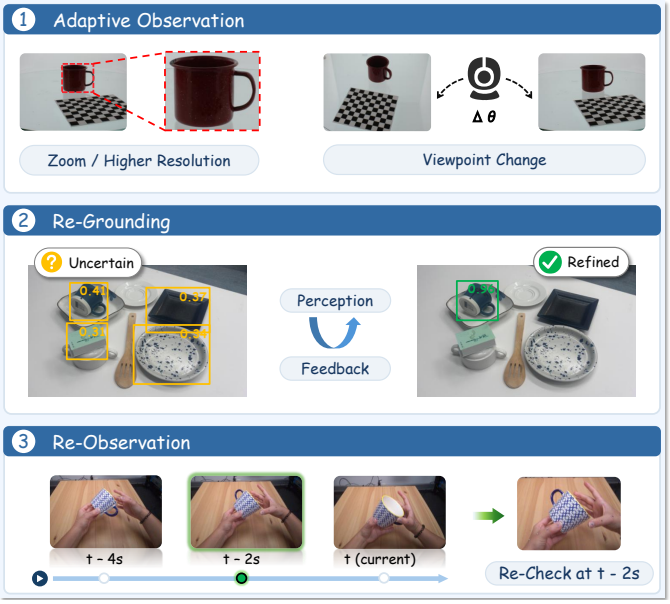


Fig. 4. **Adaptive perception through evidence refinement.** Adaptive observation (1) obtains additional evidence from a higher-resolution crop or a new viewpoint, with $\Delta\theta$ denoting the change in viewing angle. Re-grounding (2) uses feedback to refine an uncertain target match, as illustrated by the higher match score for the red mug. Re-observation (3) revisits the frame at $t-2$ s to recheck the mug.

A. Adaptive Perception

An observation that supports one decision may leave the next unresolved. A more informative view can supply missing evidence, and feedback can help the agent check whether the evidence points to the intended target. If later observations raise new questions, the agent can return to earlier evidence for another look. Fig. 4 illustrates these adjustments.

Adaptive Observation. To obtain the evidence needed for the current decision, the agent can adjust what it inspects or how closely it looks. Doc-V* [41] actively fetches cached high-resolution page tokens after inspecting thumbnails. Look Less, Reason More [193] learns when to invoke selective pixel operations. Reason with Thumbnails, Answer with Focus [194] reasons over thumbnails before requesting focused crops and accesses higher-resolution evidence when thumbnails are insufficient. GUI-Eyes [48] selects targeted crops before grounding and reasons over the enlarged views. Simple-VGC [195] interleaves visual reasoning with region focus, zooming, and reinjection of the original image. ActiveVLA [44] selects task-critical virtual views from a point cloud. SaPaVe [47] instead controls a physical camera to expose useful evidence for manipulation.

Re-Grounding. A clear view can still leave the target uncertain. The agent can use feedback to check which object or region corresponds to the target or to refine its location. DeepScan [45] refines localized evidence views through feedback between a vision-language model and visual experts. DRS-GUI [42] searches alternative interface regions and revises its focus using region-quality feedback. Visual Self-Refine [49] corrects chart-point anchors rendered on the image before decoding the corresponding values. For tracking, Dual-Agent [82] verifies target identity against temporal visual context and selectively re-localizes a target after drift or missed reappearance.

TABLE I
WITHIN-TASK ADAPTATION MECHANISMS IN VISUAL AGENTS.

Dimension	Mechanism	Target	Key Operation	Representative Works
Adaptive Perception	Adaptive Observation	Visual Input	Crop / Zoom / Select Viewpoints	[41], [44], [48]
	Re-Grounding	Visual Correspondence	Refine Entity / Region Matches	[42], [45], [82]
	Re-Observation	Prior Observations	Revisit Intervals / Views	[43], [50], [51]
Feedback-Driven Planning	Goal and Subgoal Revision	Intermediate Objectives	Revise / Add Subgoals	[14], [53], [54]
	Local Plan Repair	Local Plan Segments	Patch Affected Segments	[55], [187], [188]
	Replanning	Remaining Plan Horizon	Recompute Remaining Plan	[56], [61], [62]
	Constraint and Feasibility Reassessment	Candidates / Planning Assumptions	Check Feasibility / Reassess Assumptions	[57], [60], [189]
Dynamic Harness Orchestration	Model and Agent Routing	Models / Agents	Select or Configure Models / Route Agents	[15], [63], [66]
	Tool and Skill Orchestration	Tool Calls / Skills	Select Tools / Switch Skills	[64], [65], [69]
	Multi-Agent Coordination	Agent Interactions	Review / Hand Off / Resolve Disputes	[67], [68], [73]
Output and Action Refinement	Textual and Structured Output Correction	Answers / Predictions	Refine Responses / Structured Outputs	[74], [75], [81]
	Visual Artifact Revision	Artifacts / Prompts / Code	Edit Prompts / Content / Code	[79], [83], [190]
	Action Correction	Proposed Actions / Chunks	Revise / Reselect Actions	[77], [78]
Execution Monitoring and Control	State Tracking	State and Progress Records	Update Records / Track Progress	[86], [90], [93]
	Monitoring	Execution Conditions	Check Prerequisites / Effects	[87], [94], [95]
	Recovery	Task-Valid State	Compensate / Roll Back / Reset	[88], [92], [191], [192]
	Termination	Continuation Decision	Accept / Continue / Stop	[59], [89]

The five dimensions correspond to the L1 categories in Section III.

Re-Observation. When a later observation leaves a question unresolved, the agent may revisit earlier evidence or inspect a familiar scene from another viewpoint. VideoTreeSearch [50] uses `zoom_out` and `shift` to revisit earlier temporal choices. LensWalk [51] links observations to timestamps and entities and resamples ambiguous intervals more densely. Video-Zoomer [43] uses temporal zoom to revisit selected intervals at a higher frame rate after a sparse overview. When retrieved visual memories provide insufficient evidence, RAVEN [46] issues another semantic, spatial, or temporal query. ForkCast [52] queries a frozen policy with candidate camera views and scores the resulting action continuations to decide whether to re-observe from a new viewpoint.

B. Feedback-Driven Planning

New visual evidence can change which subgoal is worth pursuing. When the objective still holds, execution feedback can reveal a step that needs repair or show that the remaining plan needs reconsideration. Observed and predicted outcomes also help the agent judge whether candidate actions are feasible and whether its constraints need revision.

Goal and Subgoal Revision. The overall goal can remain the same as the agent replaces or adds subgoals in light of new evidence. Inner Monologue [53] incorporates updated object detections and success feedback into its language planner to revise manipulation subgoals as initially occluded objects become visible. PreAct-Nav [54] compares real observations with expected visual cues and replaces the active navigation subgoal when execution feedback indicates completion or contradicts the current objective. For image generation and editing, GenArtist [14] inspects generated images and adds corrective editing subgoals to its planning tree to address missing objects, incorrect attributes, or spatial errors.

Local Plan Repair. Observed or anticipated difficulties with part of a plan can prompt the agent to revise the affected steps and continue the task. ActionEngine [187] includes a Patcher that repairs faulty code or re-grounds failed GUI actions within the current plan separately from memory updates retained for future tasks, with the Patcher disabled in the main evaluation. VLA-Corrector [55] replaces stale remaining actions with a replanned chunk after persistent visual-dynamics disagreement. TCoT [188] checks task completion after a prescribed period of fixed global guidance and, if the task remains unfinished, generates local gripper trajectories and actions for recovery. SeqWalker [58] checks observed subinstruction progression against the instruction order, backtracking and trying an alternative action when the check fails. ForkCast [52] compares repaired and nominal arm trajectories at the same prediction horizon and adopts the repair if it yields more progress.

Replanning. If new observations or predicted outcomes favor a different course of action, the agent can revise its remaining steps. In visual model-predictive control, Sparse Imagination [56] uses sparse visual tokens to replan from fresh observations. PILOT [61] combines video-derived visual subgoals, policy priors, and a world model to refine action sequences through receding-horizon planning. IMPLEMENT [62] refines decisions by comparing predicted object-centric futures. R-WoM [196] grounds candidate computer-use simulations in retrieved tutorials to inform planning. V-ABS [197] revises beam-search preferences using executed visual-tool outcomes and entropy-weighted observer feedback. For image generation, Talk2Image [59] uses evaluation feedback to revise its intent representation and task graph, then reschedules affected generation or editing branches for re-execution.

Constraint and Feasibility Reassessment. Observations and predictions help test candidate actions against task constraints. A failed check may warrant a different action or a revision to the planner's assumptions. VLMFP [60] revises PDDL action rules when symbolic execution disagrees with consequences predicted by a visual simulator. TAMPAgent [57] couples visual belief updates with grasp, placement, and motion checks whose failures can trigger alternative candidates, replanning, or a revised scene hypothesis. APIVOT [189] checks spatial constraints and remaining free space in imagined visual states while constructing a candidate plan.

C. Dynamic Harness Orchestration

Deciding what to do next also raises the question of which capabilities the harness should invoke. Evidence from one operation may reveal the need for a specialist or a different tool or skill. When agents contribute separate results, further review can help them resolve disagreements or identify work to return for correction.

Model and Agent Routing. Which model or agent is suited to the next operation can change as observations and intermediate results arrive. MATA [63] selects the next visual reasoning agent using tool outputs and verification feedback in shared memory and reroutes after agent failures. GenMAC [15] selects a specialized correction agent based on consistency, temporal, or spatial errors detected in generated videos. VLA-Pro [66] uses current visual observations and execution history to retrieve and fuse task adapters per action chunk, restoring the base model after each chunk.

Tool and Skill Orchestration. The agent can use the evidence available at each step to select and combine tools or skills. When feedback shows that a skill no longer fits the task state, it can switch skills or return to direct action. AdaReasoner [64] conditions successive tool selections on intermediate observations, composing perception, image manipulation, and calculation tools within a visual reasoning trajectory. Earth-Agent [71] incorporates visual, spectral, and quantitative tool outputs into its interaction history to guide subsequent tool choices and arguments for Earth observation tasks. MMSkills [69] allows the agent to act directly or consult a multimodal skill in a temporary reasoning branch, which aligns reference evidence with the live state and returns procedural guidance. KnowAct-GUIClaw [70] checks whether retrieved GUI skills apply and lead to expected states, with direct GUI actions as a fallback when state checks fail. AgentVLN [72] selects perception or navigation skills from current visual observations and history, temporarily using fine-grained corrective actions when no suitable visual waypoint is available. SkillNav [65] uses current visual observations and past actions to identify the active navigation subgoal and select its specialized skill policy.

Multi-Agent Coordination. When agents disagree on a result or find an error, they may need to review each other's work to reconcile their findings or identify a step to return for correction. MACT [67] uses a judgment agent to return erroneous planning or execution steps to the responsible agent for correction before forwarding verified results to the answer agent. ORCA [68] triggers stress testing when thinker and specialist answers disagree, escalating to thesis–antithesis debate and adjudication if the specialist fails the checks. A4VL [73] uses cross-review to narrow the team and start another exploration round when agents disagree on a video question.

D. Output and Action Refinement

Visual evidence and evaluation feedback can reveal where an initial output falls short of the task requirements and guide its revision. Likewise, changes in the observed state can lead the agent to revise an upcoming action.

Textual and Structured Output Correction. Comparing an initial answer or a structured prediction with visual evidence can reveal errors. Given an initial answer, Draft and Refine [74] obtains visual-expert evidence, renders it onto the image, and re-queries the model to select a response with greater measured visual dependence. PixelCraft [81] reviews an initial answer's reasoning and tool trace, then uses identified errors to guide a second answering attempt. Visual Self-Refine [49] iteratively corrects pixel anchors on charts before extracting the corresponding values. For structured detections, AgentDet [75] uses its judge role to refine proposed object boxes. Dual-Agent [82] corrects the target visibility flag and bounding box after its verifier rejects the current tracking hypothesis, then re-prompts the tracker with accepted corrections.

Visual Artifact Revision. Inspecting a generated artifact may reveal a defect the agent can correct with a direct edit. Feedback can also guide revisions to the prompts or code that produced it. Prompt-based methods revise prompts to address missing image concepts or incorporate visual, audio, and contextual critiques of generated videos [80], [83]. Direct artifact correction includes inpainting for localized distortions [76], pose refinement using visual correspondence and physical-plausibility checks [84], simulated-audience-guided revisions to film editing and audio-visual rhythm [85], and successive image edits guided by visual verification [198]. For chart code, MM-ReCoder [79] learns to correct code over multiple turns using execution feedback. CoDA [199] guides revisions through debugging and self-reflection. WebGen-Agent [190] uses screenshot critiques and GUI tests to refine webpage code and select the best code state.

Action Correction. A failed action or a change in the observed state can prompt the agent to adjust an upcoming movement or select a different sequence. ViFailback [77] transforms manipulation failure diagnoses into textual and visual-symbolic guidance for corrective movements. After asynchronous inference, RACE [78] uses the robot's measured position and velocity to select an action chunk the robot can follow smoothly, then retimes it under kinodynamic constraints.

E. Execution Monitoring and Control

Tracking changes in the environment and task progress helps the agent assess proposed actions and check their effects. These checks can reveal when recovery is needed, and evidence about the current result helps the agent decide whether to continue or end the attempt within the remaining budget.

State Tracking. New observations and action outcomes keep the agent's record of the environment and task progress up to date. Embodied Perception Memory (EPM) [93] updates a textual environment representation from egocentric images, allowing the planner to account for changes in object locations and states. OctMem-Agent [90] updates a 3D octree map with successive RGB-D observations, recording explored regions and unexplored frontiers for subsequent aerial navigation decisions. ATMem [86] updates task records from GUI observations and operation outcomes, marking data items as pending, completed, or skipped to guide subsequent actions.

Monitoring. Before acting, the agent can check whether a proposed action satisfies the conditions for execution. Afterward, observations help it verify the intended effects and check whether task constraints still hold. At deployment, the

trained OS-Oracle critic [91] checks proposed actions and allows bounded regeneration. MaDS [200] compares competing assessments of a proposed action before execution. TaskVM [87] separates execution from verification of desired outcomes, preservation constraints, and observed state. VeriGUI [94] compares expected effects with new screenshots and uses detected failures for corrective reasoning. Q-Reflector [95] checks generated action pre- and post-conditions against paired screenshots, distinguishing execution success from task validity. Self-Grounded Verification [201] evaluates behavior against priors elicited without inspecting the candidate trajectory and uses these judgments to guide refinement. OS-Sentinel [202] combines formal trace checks with a contextual visual judge and evaluates detection primarily on frozen trajectories, with a comparison to live execution.

Recovery. A failed action can leave the environment in a state that prevents the next step. Recovery restores the conditions the agent needs to resume the task. LongHorizonUI [88] attempts local compensation before rolling back to a committed snapshot. TaskVM [87] distinguishes reversible, compensatable, and irreversible effects when supporting user-governed rollback. REACH [191] combines rollback with negative action prompts under assumptions of reversible transitions and sufficiently early detection. FLARE [92] switches from task execution to a learned reset skill after a state-breaking failure, restores task-valid conditions, and then returns control to the task policy. CoRe [192] uses counterfactual visual rollouts after an execution deviation to select which displaced objects need restoration, then restores the selected objects and realigns the robot before resuming the frozen task policy.

Termination. The agent can end an attempt once the evidence is sufficient to accept the result. If another step could resolve uncertainty or improve the result, the remaining budget determines whether the agent can continue. A4VL [73] uses cross-review and relevance ranking to accept consensus or initiate further exploration. MSSR [203] removes redundant spatial evidence and requests missing information until an adequate set supports the task. Proof-of-Perception [89] uses calibrated conformal uncertainty to decide whether to stop or gather more evidence within a compute budget, with execution certificates guiding any further evidence gathering. Draft and Refine [74] uses measured visual utilization as a grounding-based stopping proxy and retains the draft when no visual expert improves that measure. Talk2Image [59] accepts the result when evaluation feedback is stable and sufficiently high or stops refining once it exhausts its retry allowance.

IV. L2 Self-Evolving Visual Agents

In L2 self-evolution, experience from the agent's own attempts guides updates to resources that it keeps and uses in later tasks, trials, or training rounds. Training can carry what the agent has learned into the models it uses later, and retained memories and procedures can change how its harness handles subsequent tasks. The outcomes of those tasks may in turn guide revisions to the harness or reveal what the agent should learn from next. Table II relates each mechanism to the experience it uses and the resources it updates.

A. Model-Level Evolution

Feedback on an agent's outputs or trajectories can supply training signals for the models it uses in later attempts, with demonstrations or labels providing additional supervision. As these updates change how the agent reasons or acts, evaluations can reveal whether it retains earlier capabilities. Experience can also refine the predictions that a world model supplies for planning or policy learning.

Model Cognition Updates. Training with reasoning and generation feedback can change how the model later interprets evidence or produces outputs. History-Aware Reasoning [100] converts incorrect episode predictions into teacher-generated correction guidelines for subsequent reasoning training. On a fixed training set, GUI-SAGE [105] samples reasoning and action candidates from its current policy, adds a self-explanation conditioned on a labeled GUI action, and scores the candidates to update the policy. Counterfactual VLA [218] selects informative rollouts and uses expert actions to obtain teacher-labeled reasoning corrections. ReLook [101] trains a code-generation model with rendered web-code outcomes and multimodal critique before deploying it with fixed weights. DeepEyes [96] optimizes interleaved visual reasoning and inspection trajectories through reinforcement learning, retaining the acquired behavior in its model. SpatialCritic [104] trains a tool-using evaluator on separately constructed spatial preference data and uses it as a fixed reward model to update image generators through reinforcement learning.

Action and Control Model Updates. Outcomes from the agent's own actions can guide policy updates that affect which actions or tools it chooses on later attempts. EvoCUA [142] uses verified rollouts and diagnosed failures to supervise later controller training. RePO-VLA [102] uses recovery episodes annotated with progress and value labels for policy training. ComputerRL [97] alternates reinforcement and supervised updates when training its GUI policy. CGL [98] combines supervised and reinforcement updates with ground-truth action rewards. VisTA [204] trains a model to select visual tools based on task outcomes. ToolCUA [106] combines supervised and reinforcement learning to train a model that selects between GUI actions and structured tool calls.

Capability Retention. Learning a new task can weaken performance on earlier ones. Comparing results across tasks before and after an update helps establish which capabilities persist and what the model has learned. Liu et al. [219] identify benefits from small replay buffers in sequential VLA manipulation learning while distinguishing performance loss from knowledge recoverable through later fine-tuning. Zhu et al. [220] observe severe forgetting under naive sequential fine-tuning across ten physical-robot tasks and substantially better retention with configured experience replay. CGL [98] addresses continual GUI learning through entropy-guided balancing of supervised and reinforcement updates and gradient-conflict control. Dream to Remember [103] trains a steering module around a frozen generalist robot policy and rehearses earlier tasks through a frozen world model, retaining changes in the steering module. Raghavan et al. [221] use perturbed instructions to show that retained robotic task success need not

TABLE II
EXPERIENCE-DRIVEN UPDATES IN SELF-EVOLVING VISUAL AGENTS.

Experience Source	Update Process	Retained Resource	Update Target	Subsequent Use	Representative Works
Outputs / Feedback	Supervised / Reinforcement Learning	Model Parameters	Model	Visual Reasoning / Generation	[96], [100], [101], [104], [105]
Own Rollouts / Outcomes	Policy Learning	Policy Parameters	Model	Action / Tool Selection	[97], [106], [204]
Interaction Rollouts	World-Model Adaptation	World-Model Parameters	Model	Later Planning / Control	[99], [205], [206]
Procedural Guidance	Behavior Distillation	Policy Parameters	Model	Action Generation	[207], [208]
Successful Trajectories	Experience Encoding	Experience Records	Memory	Execution Guidance	[109], [111], [209]
Errors / Critiques	Guideline Induction	Diagnostic Guidance	Memory	Decision Guidance	[107], [114], [125], [210]
User Interactions	Knowledge Encoding	User Knowledge	Memory	Personalized Guidance	[112], [211]
Feedback on Memory Use	Policy Revision / Validation	Retrieval Policy	Memory	Evidence Selection	[108], [212]
Successful Executions	Procedure Extraction	Callable Procedures	Skills / Tools	Procedure Reuse	[118], [126]
Skill Failures	Reflective Revision	Skill Guidance	Skills / Tools	Later Execution	[122], [213], [214]
Tool-Use Traces	Tool Abstraction	Callable Tools	Skills / Tools	Tool Reuse	[123], [127]
Execution Feedback	Prompt / Input Revision	Prompts / Input Code	Orchestration	Later Model Inputs	[129], [131], [138]
Tool / Task Outcomes	Workflow / Control Updates	Graphs / Code / Scores	Orchestration	Execution Control	[130], [136], [215]–[217]
Rollout Signals	Data Acquisition / Refinement	Training Examples	Data / Tasks	Later Training	[139]–[141], [145]
Task Outcomes	Task / Curriculum Updates	Tasks / Curricula	Data / Tasks	Later Practice	[26], [143], [144], [148], [149]

Retained Resource identifies the specific resource kept after the update, and *Update Target* gives the corresponding L2 category. The retained resource is used in a later task, a separate attempt at the same task, or a subsequent training round.

preserve reliable language grounding.

World-Model Evolution. Observations from exploration and task execution can guide updates to world models used in later planning or control. WorldAgen [99] uses exploratory transitions to adapt the shared world-model backbone for later trials in the same scene while keeping its action-prediction head fixed. SimDist [205] updates a pretrained latent dynamics model from on-policy robot rollouts between batches of real-world trials and uses the revised model for subsequent planning. Zhou et al. [206] evaluate continual world-model learning by updating a shared dynamics backbone through online interaction and reusing it across successive simulated manipulation tasks.

B. Memory-Driven Capability Evolution

A stored record lets the agent use experience beyond the interaction that produced it. As new records accumulate, how the agent selects and organizes them affects what it can retrieve and apply to a later decision.

Experience Acquisition and Reflection. What the agent records from an interaction shapes the evidence it can use later. Reflection can turn that evidence into guidance for execution. Dejavu [111] retains visual keys and actions from rollouts whose final states meet a goal-similarity threshold. Echo [209] preserves successful trajectories with provenance identifying their source environment and model version. AtlasVA [113] uses privileged simulator state to construct its atlas and updates spatial heatmaps, visual exemplars, and symbolic skills from trajectory statistics to guide later actions and shape rewards. ViLoMem [107] uses ground-truth-assisted error attribution to develop separate visual and logical diagnostic guidelines. RSMeM [114] stores constraints distilled from failed tool-use traces in memory and retrieves them for later attempts at the same instance. DigimonGPT [117] retains procedural memories from successful tool workflows. Agent-sama [222] stores interface-state graphs and action records from completed interactions for later tasks. PerTouch [112] retains confirmed editing parameters alongside scene semantics. Kwon et al. [211] retain personalized object meanings and routines in the agent's memory. OctoT2I [115] builds a persistent knowledge base through exploratory image generation and scoring and reuses the resulting tool-capability profiles and exploration records to select generators for later requests.

Memory Management. A new record can reinforce an earlier one or bring it into question. Screening and consolidation determine which experience remains available for reuse as the store grows. ViLoMem [107] merges related diagnostic guidelines or creates new schemas. MineEvolve [125] evaluates candidate procedures and remedies for completeness, applicability, executability, specificity, and conflicts before retaining them. HyMEM [109] consolidates successful trajectories into persistent strategy and attribute graphs while keeping refreshed task-local guidance separate. Conflict-Aware Memory [116] adds successful robot experiences to its knowledge base and prunes retrieval links that may lead to conflicts. ECHO [110] commits validated trajectories to a long-term hierarchy and restructures it through idle-time clustering and consolidation. MAGNET [223] retains new interface appearances and successful workflow abstractions, with access governed by frequency and recency. ManimAgent [210] stores explanations of successes and failures alongside rendered keyframes and uses scores from a fixed VLM to select records for reuse across animation tasks.

Memory Retrieval and Reuse. The current goal and visual evidence help the agent retrieve applicable records and reuse them to guide execution. ViLoMem [107] retrieves acquired guidelines through image matching and question-conditioned reranking. HyMEM [109] expands visually and textually matched nodes through its experience graph. DigimonGPT [117] adapts replay of retained procedural experience to question complexity. PerTouch [112] retrieves scene-conditioned editing preferences for a new image. In an eight-split OpenEQA run, ABot-AgentOS [108] proposes candidate rules from memory failures and checks for gains and regressions, retaining a single retrieval rule that prioritizes observed object records by room and last-seen evidence for reuse in later splits.

C. Skill and Tool-Level Evolution

Agents can distill reusable skills from demonstrations and prior interactions or develop new skills and tools when existing

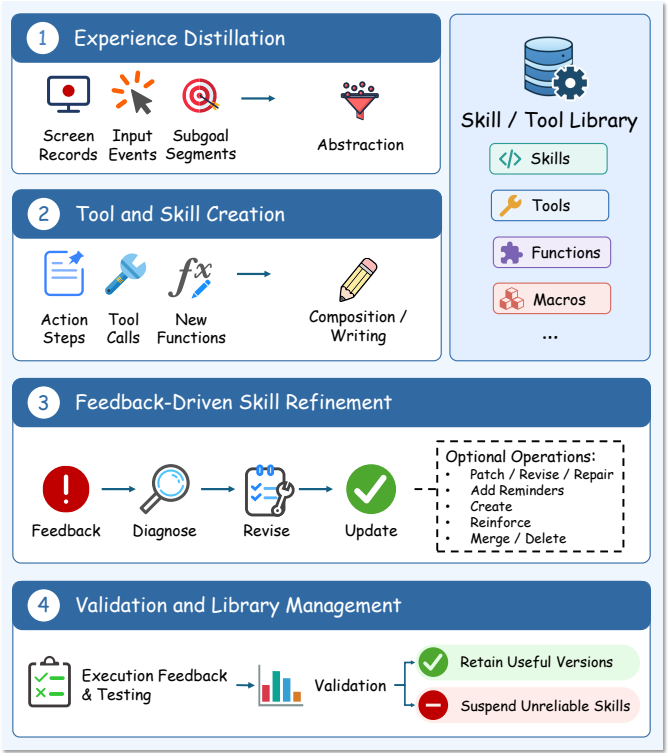


Fig. 5. **Skill and tool evolution across tasks.** Agents populate a shared library with skills distilled from task traces (1) and with new skills and tools built by composing action steps and tool calls or writing functions (2). Execution feedback helps agents diagnose shortcomings and refine skills (3). Validation uses execution feedback and tests to identify useful versions to retain and unreliable skills to suspend (4).

resources fall short. As Fig. 5 illustrates, storing these resources in a library enables reuse across tasks. Feedback from subsequent executions can then guide revisions, while validation helps determine which versions to retain.

Experience Distillation. A task trace can contain a procedure useful beyond the instance that produced it. Distilling that procedure provides guidance the agent can retrieve later or incorporate into policy training. Work2Skill [119] derives parameterized GUI guidance by aligning human screen recordings, input events, and accessibility evidence, with static checks for structural consistency and provenance during induction. ArtisanCAD [124] distills expert CAD recordings, macros, and notes into parameterized skills with operation dependencies and verification rules for later execution through CAD tools with multi-view visual feedback. MineEvolve [125] converts successful subgoal segments identified from visual observations into procedures with explicit triggers and checks, while repeated failures inform contextual remedies. SkillLens [207] represents GUI procedures and interface views as visual skill cards and distills teacher behavior conditioned on these cards into a student policy that acts without retrieval. Skill-Guided Continuation Distillation [208] extracts continuation guidance from GUI trajectories and uses skill-guided recoveries from policy-visited states to train the action model.

Tool and Skill Creation. Recurring action sequences or gaps in the available tools can motivate new executable procedures. Saving those procedures in a library makes them available to later attempts. Transductive Visual Programming [118] abstracts recurring solved spatial-reasoning programs into parameterized library functions. META [123] derives reusable macro-tools from recurring long-video tool trajectories. During training, MACRO [127] registers frequent composite tool sequences from medical-imaging rollouts whose final answers match ground truth, making them available to subsequent rollouts. EchoPath [126] converts validated GUI trajectories into callable memories with parameters, application-state preconditions, visual targets, and validation provenance. ReSight [128] distills exploration of robot-observation memories into executable builders and query tools that can be mounted for later tasks. VideoResearcher [224] implements perception tools in response to long-video reasoning failures and retains tested designs for later evidence acquisition. Agent S4 [121] builds a persistent repository of executable computer-use interfaces and usage knowledge around a fixed task model and harness.

Feedback-Driven Skill Refinement. Reusing a skill can reveal situations its current procedure handles poorly, giving the agent feedback on what to revise and retain for later tasks. VisualSkill [214] analyzes failed training trajectories to identify underrepresented UI states, explores the corresponding interfaces, and patches persistent topic guides with the resulting visual evidence. ComfyClaw [120] clusters image-generation traces to propose revisions to its skill library. In EmbodiedBench [225] tasks with visual observations, EmbodiSkill [122] distinguishes procedural defects from execution lapses, revising the retained skill body for the former while appending reminders for the latter. EvoSkill-GUI [213] revises multi-file GUI skills using immediate execution feedback and separate failed-trajectory diagnosis. Persistent Skills [226] uses desktop trajectories and evaluator feedback to update a versioned library between iterations, with each iteration executing against a frozen snapshot. Designer-RSI [186] uses recurring user requests and graded executions to acquire or repair design procedures under fixed proposal and matched replay procedures, retaining updated skills for later design tasks.

Validation and Library Management. The library's contents can change as tests and execution feedback reveal when a skill or tool works reliably. These results can guide which versions to adopt or suspend. Transductive Visual Programming [118] rewrites existing programs to use a candidate function, runs them, and checks the resulting answers before accepting the function. ComfyClaw [120] compares revised and existing libraries on synthesized held-out prompts and keeps revisions only when they show no measured performance drop. Reflect After Failure [227] checks the reliability and evidence validity of procedures alongside task outcomes and efficiency before suspending unreliable skills. Agent S4 [121] separates construction trajectories from validation tasks that the interface-building agent cannot see, then uses validation performance to select the repository to deploy. VisualEvolve [228] screens revised visual-operation libraries for performance drops relative to saved versions on a fixed held-out set.

D. Orchestration-Level Evolution

Execution feedback can help the harness decide how to proceed within a task, as discussed in Section III-C. The agent

can also use that feedback to revise the instructions and programs that shape what the model receives and how the harness directs execution, then retain those revisions for later tasks.

Prompt and Instruction Revision. When task outcomes reveal shortcomings in what the model receives, the harness can revise the guidance it supplies or the procedures that assemble its evidence. Retaining the revised context rules changes what the model sees on later attempts. EvoGen-Harness [131] revises visual requirements and constraint priorities for frozen image generators, comparing guidance versions under matched stochastic generation conditions and checking for unwanted changes on held-out cases. GeoForge [229] turns Earth-observation task traces into reusable corrections and procedural constraints, excluding unsupported workflows and answer shortcuts tied to individual samples. In visual embodied tasks, SHAPER [132] uses rollout critiques to revise procedural guidance and then context-construction code, retaining versions selected by validation for held-out execution. EnactOpt [138] uses action feedback to jointly revise execution guidance and how evidence is assembled for the planner. In video reasoning, revising the programs that select and process observations can change what visual evidence reaches the model. VideoHarness-RSI [129] uses a fixed proposer and a development-accuracy criterion to search for programs that retrieve and pack evidence or invoke auxiliary calls around a frozen VLM, retaining accepted programs for later video question answering and further refinement. Video-RSI [134] uses a fixed Editor to revisit training videos and diagnose failures before revising observation processing and memory, then applies a fixed rule based on answer accuracy and visual cost to select revisions for later reuse and refinement.

Workflow and Control Evolution. A pattern in execution outcomes can reveal where the harness needs a different rule for taking the next step. Retaining such changes can affect individual decisions or reorganize a larger part of the workflow on later attempts. VEvo-Harness [136] updates operation scores from rendered outcomes and retains verified repair patterns for later rewriting, resampling, and tuning, while keeping its routing formula and acceptance thresholds fixed. AgentCanvas [215] revises typed execution graphs from simulator outcomes and logs, but held-out reversals and the exclusion of an evaluator-leaking candidate limit the strength of its generalization claims. InstAgent [137] retains revised behavior trees with visual checks and recovery branches at inter-robot handoffs following delivery failures. RegenHarness [230] specifies a protocol that proposes changes to context rules, task templates, skill routing, and recovery policies from robot mission records, with fixed regression checks and release authorization governing adoption for later missions. LVMH [212] searches for executable memory policies that revise schemas, retrieval, and visual-evidence routing while retaining development records for later revisions. Continual Harness [231] creates, revises, and removes sub-agents and skills from gameplay traces for later bootstrap runs. EvoGen-Harness [131] revises routing, verification, retry, and termination rules around a fixed deterministic base controller. SERA [232] retains planning-context and control revisions selected from rollouts while using a fixed VLA executor. Evaluation can help determine which revisions to retain for later tasks. Zetta [133] keeps its action-selection model and orchestrator fixed while revising critics, recovery playbooks, and tools, and validates the resulting changes through diagnostic replay, fresh closed-loop execution, and multi-seed consolidation. GUI-HARVEST [130] uses repeated failures grounded in action traces and before-and-after screenshots to generate runtime-code revisions, checks scores and predicted behavior before adoption, and transfers the frozen revised harness to a second desktop benchmark. Scale and Selection [135] adopts a revised harness only when it outperforms the current harness on the same training rollout cases. KPHarness [216] selects probes using a fixed rule inspired by upper confidence bounds and proposes gameplay-harness revisions from accumulated probe utilities and visit counts, with paired validation determining which edits to retain. HarnessMM-V [217] uses task success and token costs to revise how a visual harness authors, retains, and dispatches executable decisions, with a fixed procedure for proposing and selecting rules for reuse.

E. Dynamic Data and Task Evolution

Outcomes on current tasks can reveal gaps in the experience available for learning. In response, the agent can refine the examples it learns from or change what it practices next as its capabilities develop.

Data Acquisition and Refinement. What the agent encounters during execution can guide which examples it collects or refines for later training. CorrectAD [139] analyzes driving-planner failures, formulates targeted data requirements, and synthesizes videos aligned with requested 3D layouts for subsequent planner training. HATS [140] collects semantically ambiguous GUI trajectories and iteratively repairs alignment between instructions and execution, with refinement feedback guiding later exploration. GenEvolve [147] stores differences between higher- and lower-reward image-generation trajectories as visual experience, supplying a privileged teacher view whose self-distillation targets are absorbed by the deployed student. In robotics, PLD [141] uses residual-RL specialists to collect recovery demonstrations from states reached by the base VLA, then distills these data into the generalist. SILVR [150] accumulates successful rollouts for visual-planner fine-tuning, using human success judgments to filter examples in its main real-robot experiments. Scene2Demo [146] critiques multi-view simulated rollouts and repairs action flows before collecting offline demonstrations that train behavior-cloning policies. RECALL [145] requests expert recovery demonstrations at states where the VLA is uncertain and uses them for fine-tuning, improving acquisition efficiency over passive demonstrations in a simulator-reset protocol on LIBERO-10 [233].

Task and Curriculum Adaptation. As an agent's capabilities develop, performance on current tasks can reveal where it would benefit from further practice. Those outcomes can inform which tasks it constructs or prioritizes in later rounds. SEAgent [26] updates a software guidebook from step-level trajectory judgments and descriptions of GUI state changes, then applies the same curriculum generator and prompt template across phases to turn the guidebook and new feedback into tasks for subsequent actor training. ACuRL [143] synthesizes

TABLE III
UPDATE TARGETS AND MECHANISMS SUPPORTING RECURSIVE SELF-IMPROVEMENT IN VISUAL AGENTS.

Work	Links	Evolving Component	Update Method	Driving Signal	Recursive Role
(a) Evolving Learning Signals and Experience					
GUI0 [151]	-	Curriculum Agent	RL (GRPO)	Rollout Success / Task Length	Generate Actor-Training Tasks
VLAW [154]		Video World Model	Diffusion Fine-Tuning	Real-Robot Rollouts	Generate VLA-Training Data
Grounded-Dreamer [159]	-	World Model / Inverse Dynamics	Post-Training	Simulated Robot Trajectories	Supply Replay-Verified Data
PhysEvo [153]	-	Trajectory-Review Tool	Tool Creation	Robot Rollout Records	Guide Later Skill Revisions
(b) Evolving Memory and Context Management					
EmbodiedRSI [160]	-	Memory Manager	RL (PPO)	Success Gain / Entropy Drop / Cost	Guide Experiments / Skill Updates
PRACTICE [161]	-	Skill Maintenance Policy	SFT / On-Policy Distillation	Executor Traces / Teacher Targets	Generate Skill-Library Edits
H^2-EMV [162]	-	Memory Relevance Rules	Rule Revision	Feedback on Forgotten Details	Guide Later Summaries / Retention
VideoEvolve [164]	-	Memory / Retrieval Policies	Alternating RL	Video QA Outcomes	Train Retriever on Rebuilt Memory
TaskMem [165]		Memory Writer Adapter	DPO	Recent Task Questions	Write Memory for Later Questions
(c) Evolving Update Mechanisms					
VideoEvoHarness [166]		Questioner	Rule Revision	Quality Checks / Solver Traces	Supply Harness-Update Tasks
CoDA [168]	-	Generation / Training Programs	Program Compilation	Rollout Diagnoses / Policy Progress	Generate Data / Guide Training
AdaRSI [171]	-	Shared Development Experience	Round-End Summarization	Development / Validation Records	Guide Diagnosis / Data Collection
HarnessPAI [172]		Repair Skill Library	Skill Distillation	Rollout Failures / Successful Fixes	Guide Later Program Revisions
RecurIR [167]	-	Task Generator / Revision Module	Rule Tests / Revision Learning	Degradation Tests / Plan Outcomes	Generate Tests / Guide Revisions
(d) Evolving Evaluation and Validation					
Co-EPG [178]	-	Grounding Model	SFT on Verified Data	Verified Plans / Action Labels	Supply Planner Rewards / Data
Agent0-VL [27]		Shared Solver–Verifier	SFT / RL (GRPO)	SFT Traces / Composite Rewards	Supply Step-Level Rewards
VeriCUA [181]		Judge Policy	SFT / RL (REINFORCE++)	Outcome Labels / Probe Use	Reward Actor with Frozen Verifier
RewardHarness [175]		Evaluation Skills / Tools	Validation-Gated Editing	Human Preference-Ranking Errors	Reward Image-Editor RL
SkillRubric [177]	-	Evaluation Rubrics	Diagnosis / Offline Validation	On-Policy Traces / Fixed Verdicts	Assign Turn-Level RL Credit

Links: Paper Code - Not identified in reviewed sources. *Recursive Role* indicates how the revised component contributes to a later update. RL = reinforcement learning, SFT = supervised fine-tuning, GRPO = group relative policy optimization, PPO = proximal policy optimization, DPO = direct preference optimization, QA = question answering, and VLA = vision-language-action.

tasks suited to the current action-selection model from environment exploration and feedback from earlier training iterations. EvoCUA [142] co-generates resource-grounded instructions and executable validators, turning verified rollouts and diagnosed failures into supervision for later controller training. SEES [144] diagnoses bottleneck skills in long-horizon VLA execution and constructs targeted simulated reinforcement-learning tasks around encountered states to update shared adapters. FIND [148] uses rolling success histories to prioritize feasible, lower-success tasks from a predefined library under a fixed sampling formula, training a residual controller around a frozen VLA. SimWorld Studio [149] raises navigation-task difficulty at a preset success-rate threshold, improving performance over fixed-difficulty training.

V. L3 Toward Recursive Self-Improving Visual Agents

A *visual agent* can identify where it needs to improve by gathering new evidence and relating it to experience retained from earlier attempts. After the agent acts on those findings, evaluating the outcome can expose weaknesses in how the update was produced and guide revisions to the mechanism before it is used to produce another update. Table III shows how visual agents reuse these revisions.

A. Evolving Learning Signals and Experience

As an agent develops new capabilities, earlier attempts may tell it less about the weaknesses that remain. Revising how it selects practice tasks or generates imagined trajectories can bring more informative experience into training. How the agent interprets the outcomes also matters, since diagnosing a failure can reveal which capability needs further revision.

GUI0 [151] uses rewards derived from rollout success rates to train a curriculum agent to favor tasks near the actor's capability boundary, then uses the revised agent to generate the next round of actor-training tasks. Marvo [157] updates a visual Challenger from tool-use benefits and reuses it to generate later training questions. SCALE [156] fine-tunes an action Selector on misunderstood screenshot transitions and uses it to gather experience for later Predictor training under a fixed Judger. PhysEvo [153] reuses a trajectory-review tool in a later round to trace a packing failure to contact with raised flaps, informing a skill revision that requires clearance before turning. RIWANav [158] refines its visual world model on a curriculum drawn from the navigation policy's action proposals and reuses its predictions to guide the next policy update. WMPO [152] alternates policy rollouts with world-model refinement in its StackThree experiment, using the revised model to generate imagined trajectories for the next policy update. VLAW [154] refines its video world model with real-robot policy rollouts and reuses it to generate trajectories for subsequent VLA training. WoVR [155] refines its action-conditioned world model on rollouts from the updated policy, then uses the revised model to generate imagined trajectories for another round of policy training. Grounded-Dreamer [159] adapts a video world model and an inverse-dynamics model on simulator trajectories, then uses the revised models to synthesize trajectories that are filtered by simulation replay before offline training of real-robot policies.

B. Evolving Memory and Context Management

A new problem can make earlier experience relevant again. The evidence available to the agent depends on which observations it retains, how it organizes them, and what it retrieves for

the current context. Revising these procedures changes which evidence the agent retains or supplies for later experiments, skill revisions, and training.

EmbodiedRSI [160] learns a memory manager from physical-experiment outcomes and reuses it to guide later experiments and skill revisions. PRACTICE [161] learns a procedural-knowledge maintenance policy from visual embodied trajectories and reuses it to revise the skill library for a frozen executor. H^2-EMV [162] revises natural-language relevance rules in response to feedback on forgotten details and uses them to summarize later robot observations and decide which records to retain. AutoMMemo [163] searches over executable memory-writing and retrieval programs, rebuilds memory after each revision, and evaluates the resulting guidance on separate tasks. VideoEvolve [164] updates its memory construction policy using question-answering outcomes from the current retriever, then rebuilds video memories with the revised policy to train the next retriever. In simulated streaming video tasks, TaskMem [165] ranks candidate memories by relevance to recent questions and tunes its writer's adapter with direct preference optimization, then uses the revised writer to build memories for later questions.

C. Evolving Update Mechanisms

An update mechanism can use diagnoses to propose update tasks, design revisions, or implement changes. Update outcomes can expose weaknesses and guide revisions to the mechanism. The agent can then reuse the revised mechanism to design or apply more targeted updates.

VideoEvoHarness [166] revises Questioner rules from video-question checks and Solver traces and reuses the Questioner to supply later harness-update tasks. CoDA [168] revises scene-generation and policy-training programs from visual rollout diagnoses and learning progress for later data and policy updates. The Trinity [169] consolidates Creator modules revised after failed world verification and reuses them to build training worlds for later Explorer harness updates. RSIgen [170] reuses updated understanding and generation roles to construct later curricula and supervision. AdaRSI [171] distills earlier development records into guidance for diagnosing persistent failures and targeting VLA adaptation in later rounds. HarnessPAI [172] distills rollout failures and successful fixes into repair skills that its coding agent retrieves to revise programs for later tasks. RecurIR [167] updates its task generator and plan-revision module using tests under targeted degradation changes, then uses the revised generator to supply counterexamples for later restoration policy updates.

D. Evolving Evaluation and Validation

Evaluation shapes improvement by determining how task outcomes inform training and which revisions the agent adopts. As new cases expose gaps in these judgments, execution results and external assessments can guide revisions to what the evaluator checks and how it interprets the evidence. Once the revised evaluator guides later updates, independent task outcomes help establish whether the resulting changes improve the agent's capabilities.

Co-EPG [178] reuses updated GUI-grounding checkpoints to supply rewards and verified data for later planner training. Agent0-VL [27] updates a shared Solver–Verifier whose critiques provide process rewards for later visual reasoning training. CineEvolve [174] reuses updated Critics to reward later Director updates and trains Critics on annotated Director rollouts. MagicGUI-RMS [176] retrains its domain-specific reward model on disagreements with a general-purpose evaluator and reuses the revised model to verify actions for subsequent GUI actor training. JarvisEvo [179] calibrates an image-editing evaluator against human scores before reusing it for editor training. RewardHarness [175] revises evaluation skills and tools from preference-ranking errors and reuses them for later image-editor training. VISA [180] expands image-conditioned verifier bindings through sandbox validation and reuses them in later data synthesis and reinforcement-learning rewards. SkillRubric [177] co-evolves actor guidance and screenshot-grounded rubrics before using the revised rubrics to supervise later policy updates. VeriCUA [181] learns what to check and when to stop from executable GUI task outcomes and probe-use feedback, then tests the frozen verifier as a reward source for subsequent actor training in matched runs. LIT [173] revises video-quality criteria and inspection procedures from production traces to guide later Director skill updates, while its meta engineer also revises its own task-selection and revision skills for use in later evolution epochs.

VI. Benchmarks and Evaluation of Visual-Agent Evolution

A *visual agent*'s task results show what it can do under specified conditions. To evaluate how it improves, we follow those results across attempts and ask what changes the agent carries forward and how revisions to its improvement process affect later updates.

A. Performance Evaluation at the Task-Level

Task success or functional correctness establishes whether the agent met the specified goal. Table IV illustrates how task-level protocols put that outcome in context by examining what the agent delivers and how it reaches that result.

Perception and Grounded Reasoning. A grounded response depends on inferring the relevant task state from visual evidence. Offline question-answering benchmarks probe visually grounded state understanding and reasoning about actions or changes over time [235]–[238]. For active evidence gathering, ActiView [234] measures answer accuracy after the agent selects image regions to inspect, while the active track of OpenEQA [240] also accounts for scene exploration steps. ActiveArena-Bench [239] tests how viewpoint control and interaction help agents acquire evidence for manipulation.

Planning, Execution, and Tool Interaction. Once an agent has a goal and evidence about its current state, evaluation follows whether its decisions carry the task through to completion. VisualWebArena [246] uses execution-based checks to score tasks that require interpreting webpage images, while MemU-Bench [247] measures whether agents complete mobile tasks

TABLE IV
BENCHMARKS AND TAXONOMY COVERAGE FOR TASK-LEVEL EVALUATION.

Benchmark	Links	Setting	Evaluation Target	Focus	Scale	Unit	Coverage
ActiView [234]	Paper / Docs, Code, Data / Evaluation Assets	Image Interaction	Task Capability	Shifting / Zooming	325	Distinct Questions	III-A
AgentVQA [235]	Paper / Docs, Code, Data / Evaluation Assets	Offline	Task Capability	Agent-State Understanding	1,000	Examples	III-A
Video-MME-v2 [236]	Paper / Docs, Code, Data / Evaluation Assets	Offline	Task Capability	Temporal Consistency	3,200	QA Pairs	III-A
ENACT [237]	Paper / Docs, Code, Data / Evaluation Assets	Offline	Task Capability	State–Action Reasoning	8,972	QA Pairs	III-A
MPR-GUI-Bench [238]	Paper / Docs, - , Data / Evaluation Assets	Offline	Task Capability	Multilingual GUI Reasoning	12,936	Samples	III-A
ActiveArena-Bench [239]	Paper / Docs, Code, Data / Evaluation Assets	Simulation	Task Capability	Active Visual Evidence	35	Tasks	III-A
OpenEQA [240]	Paper / Docs, Code, Data / Evaluation Assets	Simulation	Task Capability	Exploration for Active QA	557	HM3D Questions	III-A
OSWorld 2.0 [241]	Paper / Docs, Code, Data / Evaluation Assets	Interactive GUI	Task Capability	Long-Horizon Execution	108	Workflows	III-B
WebArena-Pro [242]	Paper / Docs, Code, Data / Evaluation Assets	Interactive GUI	Task Capability	Multimodal Web Tasks	300	Test Tasks	III-B
Ego2Web [243]	Paper / Docs, Code, Data / Evaluation Assets	Interactive GUI	Task Capability	Egocentric Web Tasks	500	Video-Task Pairs	III-B
NaturalGAIA [244]	Paper / Docs, Code, -	Interactive GUI	Task Capability	User-Intent Satisfaction	276	Tasks	III-B
ProSoftArena [245]	Paper / Docs, Code, Data / Evaluation Assets	Interactive GUI	Task Capability	Professional Software	436	Tasks	III-B
SuperAPP Benchmark [151]	Paper / Docs, - , -	Interactive GUI	Task Capability	Super-App Task Execution	230	Tasks	III-B
VisualWebArena [246]	Paper / Docs, Code, Data / Evaluation Assets	Interactive GUI	Task Capability	Visually Grounded Web Tasks	910	Tasks	III-B
MemU-Bench [247]	Paper / Docs, Code, Data / Evaluation Assets	Mobile GUI	Task Capability	Execution from User History	118	GUI Tasks	III-B
AsgardBench [248]	Paper / Docs, Code, Data / Evaluation Assets	Simulation	Task Capability	Visual Plan Adaptation	108	Task Instances	III-B
DeepPHY [249]	Paper / Docs, Code, -	Simulation	Task Capability	Physical Planning / Action	6	Envs.	III-B
PokeAgent (Speedrunning) [250]	Paper / Docs, Code, -	Simulation	Task Capability	Long-Horizon RPG Planning	15	First-Gym Milestones	III-B
ALFRED [251]	Paper / Docs, Code, Data / Evaluation Assets	Simulation	Task Capability	Household Goal Completion	8,055	Expert Demos	III-B
OSWorld-MCP [252]	Paper / Docs, Code, Data / Evaluation Assets	Interactive GUI	Task Capability	Tool Invocation / Outcomes	361	Evaluated Tasks	III-C
AgentVista [253]	Paper / Docs, Code, Data / Evaluation Assets	Tool Environment	Task Capability	Long Tool Workflows	209	Tasks	III-C
EgoBench [254]	Paper / Docs, Code, Data / Evaluation Assets	Tool Environment	Task Capability	Video / Tool / User Goals	1,590	Tasks	III-C
3DCodeBench [255]	Paper / Docs, Code, Data / Evaluation Assets	3D Modeling	Task Capability	3D Code Refinement	212	Object Categories	III-D
3DHarnessBench [256]	Paper / Docs, Code, Data / Evaluation Assets	3D Modeling	Task Capability	View / Geometry Access	100	Objects	III-D
PPT-Eval [257]	Paper / Docs, Code, Data / Evaluation Assets	Interactive GUI	Task Capability	Slide Creation / Editing	120	Tasks	III-D
WeGenEditBench [258]	Paper / Docs, - , -	Offline	Task Capability	Editing Constraint Checks	2,740	Eval. Instances	III-D
RefCraft [259]	Paper / Docs, Code, Data / Evaluation Assets	Tool Environment	Task Capability	Reference-Guided Webpages	300	Tasks	III-D
GenEvolve-Bench [147]	Paper / Docs, Code, Data / Evaluation Assets	Tool Environment	Task Capability	Tool-Assisted Generation	594	Held-Out Prompts	III-D
ComfyBench [260]	Paper / Docs, Code, Data / Evaluation Assets	Tool Environment	Task Capability	Visual Workflow Outputs	200	Tasks	III-D
VAB-CSS [261]	Paper / Docs, Code, -	Tool Environment	Task Capability	Target Screenshot Match	165	Test Cases	III-D
VPI-Bench [262]	Paper / Docs, Code, Data / Evaluation Assets	Interactive GUI	Task Capability	Visual Prompt Injection	306	Test Cases	III-E
CUA-Recovery [263]	Paper / Docs, Code, -	Interactive GUI	Task Capability	Inherited-Error Recovery	2,250	Error States	III-E
ProBench [264]	Paper / Docs, - , -	Mobile GUI	Task Capability	Process-Sensitive Outcomes	> 200	Tasks	III-E
FailBench [265]	Paper / Docs, - , -	Offline	Failure Detection	Robot Task-Success Judging	2,197	Attempts	III-E
SAFE-VERSE [266]	Paper / Docs, - , -	Simulation	Task Capability	Safe Execution	1,350	Hazardous Tasks	III-E
IS-Bench [267]	Paper / Docs, Code, Data / Evaluation Assets	Simulation	Task Capability	Safe Action Ordering	161	Eval. Scenarios	III-E
RecoverBench [268]	- , Code, Data / Evaluation Assets	Simulation	Task Capability	Manipulation Recovery	1,360	Error Scenes	III-E
FRBench-Sim [102]	Paper / Docs, Code, -	Simulation	Task Capability	Post-Failure Recovery	10	Evaluated Tasks	III-E
Safety-CHORES [269]	Paper / Docs, Code, Data / Evaluation Assets	Simulation	Task Capability	Cumulative Safety Cost	3	Task Families	III-E

Links Paper / Docs Code Data / Evaluation Assets. - = Unverified Public Availability. *Survey Coverage* III-A Adaptive Perception III-B Feedback-Driven Planning III-C Dynamic Harness Orchestration III-D Output and Action Refinement III-E Execution Monitoring and Control. QA = question answering, Envs. = environments, and Eval. = evaluation.

whose instructions require context from a user's earlier screenshots and actions. Other GUI and embodied benchmarks check the resulting state or apply task-specific success criteria [108], [151], [241]–[245]. ALFRED [251] reports both overall household task success and the fraction of goal conditions met. Other long-horizon benchmarks track milestones along a trajectory or test how agents adapt to changes during execution [248]–[250]. ProBench [264] also checks whether required actions occurred when the final GUI state alone cannot establish success. Tool-centered evaluations pair task outcomes with analysis of tool selection and use [252]–[254].

Artifact and Output Quality. When the task produces an artifact, its quality depends on how well it meets the user's requirements. Benchmarks for slides, webpages, and images assess content, appearance, or preservation of requested elements through rubrics and judgments of the resulting artifacts [147], [257]–[259]. ComfyBench [260] checks whether an agent-designed workflow runs and whether its image or video satisfies the request. In VisualAgentBench [261], VAB-CSS scores repairs by the structural similarity of the final webpage rendering to its target screenshot. For 3D generation, 3DCodeBench [255] scores objects built by executing code, while 3DHarnessBench [256] compares reconstructions from one or more fixed views, agent-selected views, or direct queries to the target geometry.

Reliability and Recovery. When an error interrupts execution, the agent must recognize the problem and find a way to resume progress. FailBench [265] measures failure detection from recorded attempts, while CUA-Recovery [263] and RecoverBench [268] assess task completion from error states. FRBench-Sim [102] compares manipulation success with and without a grasp disturbance during execution.

Safety and Security. Whether an agent completes a task safely depends on the actions it takes and how it responds

TABLE V
BENCHMARKS AND TAXONOMY COVERAGE FOR SELF-EVOLUTION EVALUATION.

Benchmark	Links	Setting	Evaluation Target	Focus	Scale	Unit	Coverage
Continual Manipulation [220]	Paper / Docs, Code, Data	Physical Robot	Continual Learning	Acquisition / Retention	10	Robot Tasks	IV-A
CCL Benchmark [206]	Paper / Docs, Code, -	Simulation	Continual Learning	Visual Dynamics / Forgetting	6	Task Suites	IV-A
LIBERO [233]	Paper / Docs, Code, Data	Simulation	Continual Learning	Manipulation / Forgetting	130	Tasks in 4 Suites	IV-A
CHORES [270]	Paper / Docs, Code, -	Simulation	Continual Learning	Isolated Forgetting	4	Task Sequences	IV-A
MemGUI-Bench [271]	Paper / Docs, Code, Data	Mobile GUI	Memory Capability	Cross-Session Memory	128	GUI Tasks	IV-B
M3-Bench [272]	Paper / Docs, Code, Data	Offline	Memory Capability	Audiovisual Recall	1,020	Videos	IV-B
SLVMBench [273]	Paper / Docs, -, Data	Offline	Memory Capability	Tutorial Learning / Recall	2,261	QA Pairs	IV-B
EgoStream [274]	Paper / Docs, Code, -	Offline	Memory Capability	Answer Obsolescence	2,250	Base Questions	IV-B
WorldMemArena [275]	Paper / Docs, Code, Data	Offline	Memory Capability	Memory Maintenance / Use	461	Multi-Session Tasks	IV-B
Mem-Gallery [276]	Paper / Docs, Code, Data	Offline	Memory Capability	Multimodal Memory Updates	1,711	All Eval. QA Pairs	IV-B
MemLeak [277]	Paper / Docs, Code, Data	Offline	Memory Capability	Deletion Residuals	300	Probes / Mode (Main)	IV-B
LMEE-Bench [278]	Paper / Docs, Code, Data	Simulation	Memory Capability	Exploration / Recall	166	Test Tasks	IV-B
EvoNav-Bench [279]	Paper / Docs, -, -	Simulation	Experience Reuse	Reuse / Outdated Scenes	776	Test Tasks	IV-B
IR2R-CE [280]	Paper / Docs, Code, -	Simulation	Experience Reuse	Cross-Episode Map Reuse	414	Navigation Tours	IV-B
EmbodiedCLBench [281]	Paper / Docs, Code, Data	Offline Visual Graph	Skill Acquisition	Visual Skill Composition	268	Task Graphs	IV-C
SEAR [282]	Paper / Docs, Code, -	Sim. / Physical	Experience Reuse	Skill Reuse / Transfer	335	330 Sim. / 5 Real Tasks	IV-C
PhysRSI-eval [283]	Paper / Docs, Code, -	Simulation	Workspace Evolution	Robot Revision / Transfer	18	Robot Tasks	IV-D
OpenRSI Index [284]	Paper / Docs, Code, Data	General RSI Suite	Inference Harness	Web Context / Video Pointing	2	Named Visual Tasks	IV-D
SelfImprove-Bench [285]	Paper / Docs, Code, -	Minecraft / Astray	Inference Harness	Cross-Game Transfer	10	Fixed Tasks (5 / Game)	IV-D
Procgen Curriculum Suite [286]	Paper / Docs, Code, -	Visual Games	Curriculum Learning	Adaptive Task Sampling	16	Games	IV-E

Links Paper / Docs Code Data / Evaluation Assets. - = Unverified Public Availability. *Survey Coverage* IV-A Model-Level Evolution IV-B Memory-Driven Capability Evolution IV-C Skill and Tool-Level Evolution IV-D Orchestration-Level Evolution IV-E Dynamic Data and Task Evolution. CCL Benchmark = Compositional Continual Learning Benchmark. Eval. = evaluation, QA = question answering, and Sim. = simulation.

to misleading inputs. Safety benchmarks use rule- or state-based checks to assess behavioral constraints and whether mitigation occurs before unsafe actions [266], [267]. Safety-CHORES [269] records both task success and the cumulative cost of safety violations as agents navigate and manipulate objects using camera observations. VPI-Bench [262] tests robustness to prompt injection through visual inputs.

B. Evaluation of Self-Evolution

A gain on the target tasks gives an initial measure of what a retained update adds. The protocols in Table V track that change across later attempts to assess both capability retention during continued learning and gains under new conditions.

Capability Acquisition. Acquisition can be assessed by comparing performance on the target tasks before and after the agent incorporates new experience. A control with a matched budget and the same evaluation tasks helps isolate the contribution of the update. Continual manipulation protocols track task success as the agent learns from demonstrations for newly introduced tasks [220]. WorldMemArena [275] checks how new observations inform later responses across sessions. On MemGUI-Bench [271], agents retry mobile tasks from the same device state, allowing task success to be compared with and without memory from earlier attempts. To isolate the effect of how demonstrations are collected, RECALL [145] compares success on LIBERO-10 [233] after fine-tuning with recovery demonstrations from high-uncertainty states or with demonstrations from task starts, holding the number of new demonstrations per task fixed. The Procgen Curriculum Suite [286] compares feedback-driven and random curricula for pixel-based game agents, measuring target-task success and the number of training updates needed to reach a performance threshold.

Capability Retention. Retesting earlier tasks during subsequent learning reveals how an update affects previously acquired capabilities. LIBERO [233] quantifies forgetting from the drop in a visual policy's success on previously learned manipulation tasks. In the CHORES benchmark of CORA [270], changes in return on earlier visually guided household tasks show how well a policy preserves them as training continues. Related protocols track changes in manipulation success or visual prediction over a task sequence [206], [220]. For memory, the expected answer can change with the world, so WorldMemArena and EgoStream test whether agents keep their answers current as observations become outdated [274], [275]. MemLeak [277] probes whether residual evidence still reveals information marked for deletion.

Generalization and Transfer. Held-out tasks and changed conditions test how far a retained update generalizes. Embodied-CLBench [281] evaluates whether skills learned on simpler tasks support unseen compositions. PhysRSI-eval [283] tests retained workspaces on new manipulation tasks, while SEAR [282] compares skill and memory reuse against runs without a library and time-matched restarts. In SelfImprove-Bench [285], the screenshot-based Minecraft and Astray tasks test whether a revised harness transfers to unseen instances of the same game or to the other game. Retained video harnesses are tested on held-out question answering [129], [134], and the MolmoWeb and Molmo2 tasks in OpenRSI Index [284] assess revised inference strategies on held-out cases. IR2R-CE [280] tests whether maps retained across routes help agents follow new instructions in the same building. Other benchmarks probe memory reuse in visual reasoning and navigation [272], [275], [278], [279]. SLVMBench [273] tests transfer from supplied video tutorials to new questions, while Mem-Gallery [276] probes memory using multimodal conversation histories.

C. Evaluation of Recursive Self-Improvement

Evidence of recursive reuse starts with tracing how a revised updater U produces a later retained update and, when

TABLE VI
BENCHMARKS AND TAXONOMY COVERAGE FOR EVALUATING RECURSIVE SELF-IMPROVEMENT.

Benchmark	Links	Setting	Evaluation Target	Focus	Scale	Unit	Coverage
Flex-RSI [287]		Vision / Robotics	Harness RSI	Recursive Harness Updates	Unlisted	Selected Tasks	V
AI4AI-Bench [288]	-	General RSI Suite	Update-Method Design	Diffusion RL / CLIP Merge	2	Visual Families	V
RSI-Exam [289]		General RSI Suite	Update-Method Design	VLM Reward / Training	1	Counting Task	V
AutoResearchExam [290]	-	Research Suite	Procedure Design	CLIP Merge / Video Caching	2	Named Visual Tasks	V
Vals RSI Index [291]	- -	General RSI Suite	Evaluator Design	Visual Web-App Judge	24	Held-Out Apps	V

Links Paper / Docs Code Data / Evaluation Assets. - = Unverified Public Availability. *Survey Coverage* V Recursive Self-Improvement and Mechanism Design. RL = reinforcement learning and VLM = vision-language model.

evolution of Γ is claimed, how the revised procedure is used to revise U again. Controlled comparisons can then estimate the contribution of those revisions, while the protocols in Table VI illustrate ways to observe different parts of the process.

RSI Benchmarks for Visual Agents. Successive evaluation rounds can show where an adopted revision takes effect in the next update. Flex-RSI [287] evaluates visual and robotics tasks over repeated executor and improver rounds, allowing the improver to revise both the executor and its own procedures for later reuse. Validation checks candidate solutions, while held-out tests measure performance under fixed budgets.

Mechanism-Design Tasks with Visual Inputs. Each general-purpose research benchmark includes visual tasks as one subset of its design problems. Scores on these visual tasks can assess the outcome of a mechanism revision, while evidence of visual RSI comes from tracing the revised mechanism to a later retained update. AI4AI-Bench [288] scores redesigned diffusion training and CLIP merging by image aesthetics and ImageNet-V2 accuracy, respectively, and RSI-Exam [289] evaluates a model trained with a redesigned reward on hidden visual counting tasks. AutoResearchExam [290] tests CLIP merges on held-out classification and scores video caching by fidelity to uncached outputs under a compute limit. Vals RSI Index [291] measures agreement between a redesigned judge and reference ratings on held-out web apps.

Downstream Visual Benchmarks. Standard visual benchmarks score the agent's performance on a defined set of tasks. In a visual RSI study, testing the initial and revised agents under the same conditions measures downstream gains, while records of a later retained update show whether the revised updater was actually used. VideoEvoHarness [166] compares multiple-choice accuracy under initial and evolved harnesses on LongVideoBench [292], MMVU [293], and Video-Holmes [294], and tracks gains across rounds on a fixed question set. For image generation, RSIgen [170] compares instruction adherence and image quality on held-out requests drawn from SearchGen [295]. AutoMMemo [163] evaluates revised memory programs on web and multimodal question-answering tasks, and EmbodiedRSI [160] measures success on unseen manipulation tasks in RoboCasa365 [296]. JarvisEvo [179] uses ArtEdit-Bench-Lr to score instruction adherence and fidelity to reference edits after evaluator-guided training. RewardHarness [175] also assesses image edits, while CineEvolve [174] scores videos from its evolved production workflow. VeriCUA [181] measures GUI task success after training an actor with the revised verifier, and SkillRubric [177] scores a policy trained with revised rubrics on held-out multimodal tool-use tasks.

VII. OPEN CHALLENGES AND FUTURE DIRECTIONS

Balancing Capability Retention and Generalization. A central goal of visual agent evolution is to retain useful visual representations, object–action associations, and task execution strategies while generalizing to new visual observations, object instances, spatial configurations, and task objectives. Prior work has explored replay-based learning, knowledge preservation, and adaptation through new task demonstrations to mitigate forgetting and acquire new capabilities [103], [219]–[221]. Relying heavily on past experience can constrain adaptation as conditions change, and updating for new tasks can erode previously acquired capabilities. An open challenge is to preserve transferable visual knowledge across successive updates while selectively adapting task-specific behaviors. Recording the visual and task conditions under which each consolidated skill applies could help agents retrieve, adapt, and recombine relevant skills as observations and objectives change, subject to practical interaction and memory budgets.

Preventing Objective Hacking in Visual Agent Evolution. A central challenge in visual agent evolution is to ensure that apparent performance improvements reflect genuine capability gains rather than exploitation of evaluation objectives, execution environments, or tool interfaces. Prior work has explored sandboxed execution, phase-specific write restrictions, and independent release checks to constrain agent modifications and detect unreliable improvements [7], [230], [287]. Objective hacking remains possible under execution isolation. In one sandboxed system, high evaluation scores masked persistent tool-use hallucinations. Even with write restrictions and release checks, agents may exploit evaluation criteria or reuse approved workflows beyond their intended task scope. An open challenge is to maintain independent evaluation and enforce task-specific authorization throughout successive evolution cycles and workflow reuse. Future safeguards could establish security boundaries outside the agent-editable workflow, independently audit agent behaviors and evaluation outcomes, and incorporate human oversight for consequential deployments.

Maintaining Consistency in Long-Horizon Execution. A central challenge in long-horizon visual agent execution is to maintain consistency between task objectives, intermediate decisions, and evolving environmental states across successive interactions. Perception errors and unexpected action outcomes can invalidate the state assumptions behind subsequent decisions. Acting on those assumptions can compound the initial error and divert execution from the task objective. Prior work has used visual observations to verify expected action effects and check whether task constraints still hold [87], [94]. An open challenge is to infer overall task progress from these checks

when observations are incomplete and the environment changes. Tracking the observations that support each task milestone could help agents reassess progress when new evidence invalidates earlier state assumptions. Agents could then update their state estimates and revise the remaining plan in line with the original task objective.

Verifying Genuine Recursive Self-Improvement. A fundamental challenge in recursive self-improvement (RSI) is to establish whether improvements across successive evolution rounds result from a genuinely improved updater or merely from accumulating experience and additional computational resources. Prior work has explored revising components such as curriculum generators and evaluators and reusing them in subsequent training rounds [151], [179]. However, showing that an updated component participates in later evolution does not establish that it improves the update process itself, as observed gains may also arise from repeated optimization. An open question is whether updater improvements yield sustained benefits across successive updates. Controlled comparisons between revised and original updaters under matched experience and resource budgets could help establish causal contributions, while tracing how revisions alter subsequent data generation, feedback, and learning dynamics could reveal the mechanisms underlying genuine recursive improvement.

Benchmark Development for Visual Agents. Task-level metrics alone cannot reveal whether an agent succeeds through reliable perception, planning, and execution or through compensating behaviors and environmental shortcuts. Prior work has explored process-level verification of required actions, controlled evaluation of retained workspaces, and multi-round assessment of successive agent revisions [264], [283], [287], [297]. However, checking intermediate actions as additional task outputs does not establish the effectiveness of individual agent components or their contributions to overall performance. An open challenge is to jointly evaluate task outcomes, component-level capabilities, and the functional contributions of harness components under comparable conditions across successive updates. A shared evaluation framework could combine task-based assessments with component-specific tests and controlled interventions, enabling systematic comparisons of capability development, component effectiveness, reliability, and resource costs across visual agents.

VIII. Conclusion

This survey systematically explores recent advances in the adaptation and evolution of visual agents, providing a unified perspective on this rapidly developing field. We first introduce essential background knowledge, covering the definition, foundations, and complementary roles of model capabilities and agent harnesses. Subsequently, we comprehensively review representative methods across visual understanding, visual generation and editing, computer use, and embodied interaction, and organize them into a technical taxonomy spanning within-task adaptation, self-evolution, and recursive self-improvement. In the evaluation section, we summarize representative benchmarks, evaluation metrics, and protocols for assessing task performance, capability development, retention, transfer, reliability, and recursive improvement. Lastly, we highlight current challenges and outline promising future directions to inspire continued research toward visual agents that can improve more reliably and generalize across tasks and environments.

References

[1] S. Yao, J. Zhao *et al.*, "ReAct: Synergizing Reasoning and Acting in Language Models," in *ICLR*, 2023.

[2] Y. Shang, Y. Li *et al.*, "AgentSquare: Automatic LLM agent search in modular design space," in *ICLR*, 2025.

[3] M. Zhang, Y. Yang, R. Xie, B. Dhingra, S. Zhou, and J. Pei, "Generalizability of large language model-based agents: A comprehensive survey," *ACM Comput. Surv.*, 2026.

[4] S. Wu, S. Zhao *et al.*, "AVATAR: Optimizing LLM agents for tool usage via contrastive reasoning," in *NeurIPS*, 2024.

[5] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: language agents with verbal reinforcement learning," in *NeurIPS*, 2023.

[6] G. Wang, Y. Xie *et al.*, "Voyager: An open-ended embodied agent with large language models," *arXiv*, 2023.

[7] J. Zhang, S. Hu, C. Lu, R. Lange, and J. Clune, "Darwin Gödel Machine: Open-Ended Evolution of Self-Improving Agents," in *ICLR*, 2026.

[8] Y. Zheng, D. Fu *et al.*, "DeepResearcher: Scaling deep research via reinforcement learning in real-world environments," in *EMNLP*, 2025.

[9] X. Wang, B. Li *et al.*, "OpenHands: An open platform for AI software developers as generalist agents," in *ICLR*, 2025.

[10] H. Luo, H. Feng *et al.*, "AgentMath: Empowering mathematical reasoning for large language models via tool-augmented agent," in *ICLR*, 2026.

[11] T. Gupta and A. Kembhavi, "Visual Programming: Compositional Visual Reasoning Without Training," in *CVPR*, 2023.

[12] D. Surís, S. Menon, and C. Vondrick, "ViperGPT: Visual Inference via Python Execution for Reasoning," in *ICCV*, 2023.

[13] X. Zhang, Z. Jia *et al.*, "Deep Video Discovery: Agentic Search with Tool Use for Long-form Video Understanding," in *NeurIPS*, 2025.

[14] Z. Wang, A. Li, Z. Li, and X. Liu, "GenArtist: Multimodal LLM as an agent for unified image generation and editing," in *NeurIPS*, 2024.

[15] K. Huang, Y. Huang, X. Ning, Z. Lin, Y. Wang, and X. Liu, "Genmac: compositional text-to-video generation with multi-agent collaboration," in *AAAI*, 2026.

[16] H. Zhou, L. Huang *et al.*, "VideoAgent: All-in-One Framework for Video Understanding and Editing," in *EMNLP*, 2026.

[17] Q. Wu, K. Cheng *et al.*, "GUI-Actor: Coordinate-Free Visual Grounding for GUI Agents," in *NeurIPS*, 2025.

[18] S. Zhou, F. F. Xu *et al.*, "Webarena: A realistic web environment for building autonomous agents," in *ICLR*, 2024.

[19] T. Xie, D. Zhang *et al.*, "OSWorld: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments," in *NeurIPS*, 2024.

[20] S. Agashe, J. Han, S. Gan, J. Yang, A. Li, and X. Wang, "Agent s: An open agentic framework that uses computers like a human," in *ICLR*, 2025.

[21] D. Hafner, T. Lillicrap *et al.*, "Learning latent dynamics for planning from pixels," in *ICML*, 2019.

[22] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to Control: Learning Behaviors by Latent Imagination," in *ICLR*, 2020.

[23] Z. Wang, S. Cai *et al.*, "JARVIS-1: Open-World Multi-Task Agents With Memory-Augmented Multimodal Language Models," *TPAMI*, 2025.

[24] J. Xie, Z. Chen, R. Zhang, and G. Li, "Large multimodal agents: A survey," *Visual Intelligence*, 2025.

[25] N. Mokaria, R. Raj *et al.*, "A survey on foundations and frontiers of multimodal agentic frameworks: Techniques and applications," *arXiv preprint arXiv:2608.20379*, 2026.

[26] Z. Sun, Z. Liu *et al.*, "SEAgent: Self-Evolving Computer Use Agent with Autonomous Learning from Experience," in *ICML*, 2026.

[27] J. Liu, K. Xiong *et al.*, "Agent0-VL: Exploring Self-Evolving Agent for Tool-Integrated Vision-Language Reasoning," in *ICML*, 2026.

[28] Z. Ren, Y. Chen *et al.*, "Self-improvements in modern agentic systems: A survey," *arXiv*, 2026.

[29] Y. Xu, W. Zhang, Y. Chen, X. Lin, and Y. Zhang, "Self-evolving agents as dynamic graph transformation: A survey and new perspective," *arXiv*, 2026.

[30] S. Liu, Z. Lin *et al.*, "The path to recursive self-improving agents: Foundation, framework, and future directions," *Preprints*, 2026.

[31] X. Yu, C. Liu, W. Long, H. Zhang, and L. Xue, "From harness optimization to recursive self-improvement: A survey of language model agents," *Preprints*, 2026.

[32] J. Jiang, "A cross-scenario survey and analysis of self-evolving multi-modal agents," *arXiv*, 2026.

[33] Z. Xing, J. Wang *et al.*, "Agentic aigc: A survey of visual generation agents," *Preprints*, 2026.

[34] Y. Huang, S. Tu *et al.*, "Agentic visual generation: From generative models to agentic control," *arXiv*, 2026.

[35] L. Fan, Y. Zheng *et al.*, "Video generation meets agent harness: Agentic video generation, agentic world modeling and recursive self-improvement," *arXiv*, 2026.

[36] P. J. Sager, B. Meyer *et al.*, "A comprehensive survey of agents for computer use: Foundations, challenges, and future directions," *J. Artif. Intell. Res.*, 2026.

[37] J. Gu, E. Stefani, Q. Wu, J. Thomason, and X. Wang, "Vision-and-Language Navigation: A Survey of Tasks, Methods, and Future Directions," in *ACL*, 2022.

[38] C. Qin, D. Wu *et al.*, "Towards agentic robots: Model, data, environment, and harness," *arXiv*, 2026.

[39] M. Luo, Z. Bai *et al.*, "Toward recursive self-improvement in physical ai," *arXiv*, 2026.

[40] X. Han, K. Luo *et al.*, "Recursive self-improvement in physical ai: Mechanisms, evidence, and evaluation," *arXiv*, 2026.

[41] Y. Zheng, P. Fu *et al.*, "Doc-V*: Coarse-to-Fine Interactive Visual Reasoning for Multi-Page Document VQA," in *ACL*, 2026.

[42] Y. Liu, H. Shen, L. Yu, S. Liu, Z. Chen, and Y. Zhou, "DRS-GUI: Dynamic Region Search for Training-Free GUI Grounding," in *CVPR*, 2026.

[43] Y. Ding, X. Lai, Y. Zhang, W. Li, R. Chu, and Y. Yang, "Videozoomer: Reinforcement-learned temporal focusing for long video reasoning," in *ICLR*, 2026.

[44] Z. Liu, Y. Gu, Y. Wang, X. Xue, and Y. Fu, "ActiveVLA: Injecting Active Perception into Vision-Language-Action Models for Precise 3D Robotic Manipulation," in *CVPR*, 2026.

[45] Y. Li, H. Zhan, J. Chen, Y. Gong, Q. Liu, and Y. Lu, "DeepScan: A Training-Free Framework for Visually Grounded Reasoning in Large Vision-Language Models," in *CVPR*, 2026.

[46] Y. Hu, Z. Zheng *et al.*, "Raven: Long-horizon reasoning & navigation with a visuo-spatio-temporal memory," *arXiv*, 2026.

[47] M. Liu, E. Zhou *et al.*, "SaPaVe: Towards Active Perception and Manipulation in Vision-Language Action Models for Robotics," in *CVPR*, 2026.

[48] C. Chen, J. Shao *et al.*, "Gui-eyes: Tool-augmented perception for visual grounding in gui agents," in *AAAI*, 2026.

[49] J. Li, X. Dong, Y. Zang, Y. Cao, J. Wang, and D. Lin, "Visual Self-Refine: A Pixel-Guided Paradigm for Accurate Chart Parsing," in *ICLR*, 2026.

[50] C. Zhang, Z. Wang *et al.*, "Searching Videos as Trees: Self-Correcting Agents for Grounded Long Video QA," in *NeurIPS*, 2026.

[51] K. Li, Y. Li, H. Shen, M. Liu, H. Chang, and S. Shan, "LensWalk: Agentic Video Understanding by Planning How You See in Videos," in *CVPR*, 2026.

[52] C. Wen, H. Zeng *et al.*, "Forkcast: Predictive recovery for vision-language-action policies via counterfactual futures," *arXiv*, 2026.

[53] W. Huang, F. Xia *et al.*, "Inner Monologue: Embodied Reasoning through Planning with Language Models," in *CoRL*, 2022.

[54] J. Xie, S. Ruan *et al.*, "Preact-nav: Agentic reasoning before action for urban navigation," *arXiv*, 2026.

[55] Y. Pan, M. Pan *et al.*, "VLA-Corrector: Lightweight Detect-and-Correct Inference for Adaptive Action Horizon," in *NeurIPS*, 2026.

[56] J. Chun, Y. Jeong, and T. Kim, "Sparse imagination for efficient visual world model planning," in *ICLR*, 2026.

[57] U. A. Mishra, A. Degirmenci, Y. Chen, D. Xu, T. Hermans, and C. Garrett, "Tampagent: Agentic belief-space planning for mobile manipulation," *arXiv*, 2026.

[58] Z. Han, X. Wang *et al.*, "Seqwalker: Sequential-horizon vision-and-language navigation with hierarchical planning," in *AAAI*, 2026.

[59] S. Ma, Y. Guo, J. Su, Q. Huang, Z. Zhou, and Y. Wang, "Talk2Image: A multi-agent system for multi-turn image generation and editing," in *AAAI*, 2026.

[60] Y. Hao, Y. Chen, C. Fan, and Y. Zhang, "Simulation to rules: A dual-vlm framework for formal visual planning," in *ICLR*, 2026.

[61] Y. Liu, H. Cui, and K. Desingh, "Pilot: Guiding long-horizon world-model planning for robot manipulation with visual subgoals and policy priors," *arXiv*, 2026.

[62] J. Song, H. Jin *et al.*, "Model-Based Imaginative Planning for Embodied Agents," in *ACL*, 2026.

[63] Z. Cai, F. Ke *et al.*, "MATA: A Trainable Hierarchical Automaton System for Multi-Agent Visual Reasoning," in *ICLR*, 2026.

[64] M. Song, H. Sun, J. Gu, L. Li, R. Krishna, and Y. Cheng, "Adareasoner: Dynamic tool orchestration for iterative visual reasoning," in *ICLR*, 2026.

[65] T. Ma, Y. Zhang, Z. Wang, and P. Kordjamshidi, "Breaking down and building up: Mixture of skill-based vision-and-language navigation agents," in *ACL*, 2026.

[66] S. Si, Y. Lu, R. Yang, Z. Ye, Z. Wu, and Y.-G. Jiang, "VLA-Pro: Cross-Task Procedural Memory Transfer for Vision-Language-Action Models," in *NeurIPS*, 2026.

[67] X. Yu, C. Xu *et al.*, "Visual Document Understanding and Reasoning: A Multi-Agent Collaboration Framework with Agent-Wise Adaptive Test-Time Scaling," in *CVPR*, 2026.

[68] A. Lassoued, M. A. Souibgui, and Y. Kessentini, "Orca: Orchestrated reasoning with collaborative agents for document visual question answering," *arXiv*, 2026.

[69] K. Zhang, S. Shao *et al.*, "MMSkills: Towards Multimodal Skills for General Visual Agents," in *NeurIPS*, 2026.

[70] Y. Li, J. Li *et al.*, "Knowact-guiclaw: Know deeply, act perfectly, personal gui assistant with self-evolving memory and skill," *arXiv*, 2026.

[71] P. Feng, Z. Lv *et al.*, "Earth-agent: Unlocking the full landscape of earth observation with agents," in *ICLR*, 2026.

[72] Z. Xin, W. Li *et al.*, "Agentvln: Towards agentic vision-and-language navigation," in *ECCV*, 2026.

[73] Y. Xu, G. Liu *et al.*, "A Multi-Agent Perception-Action Alliance for Efficient Long Video Reasoning," in *CVPR*, 2026.

[74] S. Jeong, R. Masukawa *et al.*, "Draft and Refine with Visual Experts," in *CVPR*, 2026.

[75] H. Li, Y. Wang, Z. Yan, L. Wen, and B. Huang, "AgentDet: A Shared-Blackboard Multi-Agent Framework for Zero-/Few-Shot Object Detection," in *CVPR*, 2026.

[76] S. Shen, J. Liang *et al.*, "Agentic Retoucher for Text-To-Image Generation," in *CVPR*, 2026.

[77] X. Zeng, X. Zhou *et al.*, "Diagnose, Correct, and Learn from Manipulation Failures via Visual Symbols," in *CVPR*, 2026.

[78] S. Kim, J. Kim, and J. J. Lim, "Time Optimal Execution of Action Chunk Policies Beyond Demonstration Speed," in *ICLR*, 2026.

[79] Z. Tang, X. Zhang *et al.*, "MM-ReCoder: Advancing Chart-to-Code Generation with Reinforcement Learning and Self-Correction," in *CVPR*, 2026.

[80] D. X. Long, X. Wan, H. Nakhost, C.-Y. Lee, T. Pfister, and S. O. Arik, "VISTA: A Test-Time Self-Improving Video Generation Agent," in *CVPR*, 2026.

[81] S. Zhang, Z. Li *et al.*, "Pixelcraft: A multi-agent system for high-fidelity visual reasoning on structured images," in *ICLR*, 2026.

[82] V. Bundele, S. Agrawal, M. Hosseinzadeh, T. N. Nguyen, and H. Lensch, "When trackers fail: Vlm-guided verification and recovery for robust video object segmentation," *arXiv*, 2026.

[83] S. Wu, M. Sun, W. Wang, Y. Wang, and J. Liu, "Visualprompter: Semantic-aware prompt optimization with visual feedback for text-to-image synthesis," in *ICLR*, 2026.

[84] L. Ling, C.-H. Lin *et al.*, "Scenethesis: A language and vision agentic framework for 3d scene generation," in *ICLR*, 2026.

[85] K. Huang, Y. Huang *et al.*, "FilMaster: Bridging Cinematic Principles and Generative AI for Automated Film Generation," in *ICLR*, 2026.

[86] C. Liu, L. Chen *et al.*, "What memory do gui agents really need? from passive records to active task-driving states," *arXiv*, 2026.

[87] W. Yang, Y. Hao *et al.*, "Task-state governance: A virtual machine for verifiable and recoverable gui agent execution," *arXiv*, 2026.

[88] B. Kang, S. Wen *et al.*, "LongHorizonUI: A Unified Framework for Robust long-horizon Task Automation of GUI Agent," in *ICLR*, 2026.

[89] A. Fayyazi and H. Akrami, "Proof-of-Perception: Certified Tool-Using Multimodal Reasoning with Compositional Conformal Guarantees," in *CVPR*, 2026.

[90] J. Zhou, J. Miao, Y. Lin, X. Wang, J. Xiao, and J. Yu, "Memory-Augmented Scene Understanding and Exploration for Open-World Aerial Object-Goal Navigation," in *CVPR*, 2026.

[91] Z. Wu, J. Xie *et al.*, "OS-Oracle: A Comprehensive Framework for Cross-Platform GUI Critic Models," in *CVPR*, 2026.

[92] G. Zhao, Z. Tang, X. Chen, Z. Kuang, Y. Tian, and G. Li, "FLARE: A Failure-Aware Framework for Autonomous Correction and Recovery in Visual-Language Robotic Manipulation," in *CVPR*, 2026.

[93] P. Parashar, J. Krantz, M. Chang, K. Shah, X. Puig, and R. Mottaghi, "Planning with an Embodied Learnable Memory," in *ICLR*, 2026.

[94] Y. Zhang, X. Xue *et al.*, "Don't Act Blindly: Robust GUI Automation via Action-Effect Verification and Self-Correction," in *ACL*, 2026.

[95] L. Guo, W. Liu, L. Gu, Y. Wang, Tse-Hsun, and Chen, "Q-reflector: Learning gui action reflection with generated conditions," *arXiv*, 2026.

[96] Z. Zheng, M. Yang *et al.*, "DeepEyes: Incentivizing "Thinking with Images" via Reinforcement Learning," in *ICLR*, 2026.

[97] H. Lai, X. Liu *et al.*, "ComputerRL: Scaling End-to-End Online Reinforcement Learning for Computer Use Agents," in *ICLR*, 2026.

[98] Z. Yao, Z. Huang *et al.*, "CGL: Advancing Continual GUI Learning via Reinforcement Fine-Tuning," in *CVPR*, 2026.

[99] C. Wan, K. Wang, Y. Si, P. Zhang, and M. Li, "Worldagen: Unified state-action prediction with test-time world model training," in *AAAI*, 2026.

[100] Z. Wang, L. Yang *et al.*, "History-Aware Reasoning for GUI Agents," in *AAAI*, 2026.

[101] Y. Li, C. Zhang *et al.*, "ReLook: Vision-Grounded RL with a Multimodal LLM Critic for Agentic Web Coding," in *ACL*, 2026.

[102] W. Liufu, X. Guo *et al.*, "RePO-VLA: Recovery-Driven Policy Optimization for Vision-Language-Action Models," in *NeurIPS*, 2026.

[103] B. Tharwat, I. Nematollahi, M. Alles, X. Zhang, A. L. Chandra, and A. Valada, "Dream to remember: Continual steering of generalist robot policies with world models," *arXiv*, 2026.

[104] X. Tan, J. Lin *et al.*, "Getting space right: Agentic rewards for image generation," *arXiv*, 2026.

[105] F. Tang, Z. Gu *et al.*, "GUI-SAGE: Enhancing GUI Automation with Self-Explanatory Learning," in *CVPR*, 2026.

[106] X. Hu, X. Zhang *et al.*, "ToolCUA: Towards Optimal GUI-Tool Path Orchestration for Computer Use Agents," in *NeurIPS*, 2026.

[107] W. Bo, S. Zhang *et al.*, "ViLoMem: Agentic Learner with Grow-and-Refine Multimodal Semantic Memory," in *CVPR*, 2026.

[108] J. Tian, S. Liu *et al.*, "Abot-agentos: A general robotic agent os with lifelong multi-modal memory," *arXiv*, 2026.

[109] S. Zhu, W. Wu, K. Zhou, S. Wang, and B. Huang, "Hybrid self-evolving structured memory for gui agents," *arXiv*, 2026.

[110] Y. Hu, J. Cui *et al.*, "ECHO: Continuous Hierarchical Memory for Vision-Language-Action Models," in *NeurIPS*, 2026.

[111] S. Wu, Y. Ji *et al.*, "Dejavu: Towards Experience Feedback Learning for Embodied Intelligence," in *CVPR*, 2026.

[112] Z. Chang, Z.-P. Duan *et al.*, "PerTouch: VLM-Driven Agent for Personalized and Semantic Image Retouching," in *AAAI*, 2026.

[113] P. Wang, Y. Hu, X. Liu, J. Yang, H. Wang, and Z. Wen, "Atlasva: Self-evolving visual skill memory for teacher-free vlm agents," *arXiv*, 2026.

[114] B. Wu, Y. Zhang *et al.*, "RSMeM: Knowledge-Enhanced Memory Evolution for Remote Sensing Agents with Systematic Evaluation," in *ACL*, 2026.

[115] X. Jiang, B. Chen, G. Li, Y. Duan, R. Wang, and J. Zhang, "OctoT2I: A Self-Evolving Agentic Text-to-Image Router," in *CVPR*, 2026.

[116] K. Ma, H. Wang, S. Chen, Y. Cai, R. Jin *et al.*, "Conflict-aware memory for embodied agents: Enhancing vector data quality via detection rules," in *ACL*, 2026.

[117] B. Li, X. Zhang, T. Liu, S. Wang, and Y. Cheng, "DigimonGPT: An Evolvable Agent with Hierarchical Human-like Memory for Video Question Answering," in *AAAI*, 2026.

[118] S. Wu, X. Wang, Y. Zhang, H. Zhu, and S. Yeung, "Transductive Visual Programming: Evolving Tool Libraries from Experience for Spatial Reasoning," in *ICLR*, 2026.

[119] Y. Huang, H. Li *et al.*, "Work2skill: Capturing human gui experience for reusable agent skill induction," *arXiv*, 2026.

[120] Z. Li, D. Liu *et al.*, "Comfyclaw: Self-evolving skill harnesses for image generation workflows," *arXiv*, 2026.

[121] S. Agashe, Y. Yang *et al.*, "Evolving Agent Interfaces," *arXiv*, 2026.

[122] R. Ju, X. Wang *et al.*, "Embodiskill: Skill-aware reflection for self-evolving embodied agents," *arXiv*, 2026.

[123] J. Huang, L. Chen *et al.*, "META: Meta Evolution of Tool Trajectory Adaptation for Long-Video Understanding," in *CVPR*, 2026.

[124] Y. Xu, Q. Wu *et al.*, "Artisancad: An industrial-level cad agent with expert-grounded knowledge distillation," *arXiv*, 2026.

[125] Z. Xie, Z. Chen *et al.*, "Mineevolve: Self-evolution with accumulated knowledge for long-horizon embodied minecraft agents," *arXiv*, 2026.

[126] Y. Zhao, A. Shanmugham, S. Roy, and Y. Xu, "Echopath: Execution-level replayable memory for gui agents," *arXiv*, 2026.

[127] L. Fan, P. Dai *et al.*, "Evolving medical imaging agents via experience-driven self-skill discovery," *arXiv*, 2026.

[128] L. Che, Y. Chen *et al.*, "Resight: Task-driven self-improving memory for embodied agents," *arXiv*, 2026.

[129] G. Xu and H. Chen, "Videoharness-rsi: Recursive harness self-improvement for long-video understanding with frozen vision-language models," *arXiv*, 2026.

[130] G. Yang, Z. Qu *et al.*, "Gui-harvest: Self-improving gui agents through evidence-driven harness evolution," *arXiv*, 2026.

[131] J. Luo, Y. Liu, C. Meng, and Y. Guo, "Evogen-harness: Learning where and how to evolve image-generation harnesses," *arXiv*, 2026.

[132] P. Wang, Z. Ma *et al.*, "Self-evolving embodied agents via skill-harness evolution," *arXiv*, 2026.

[133] X. Ding, L. Mi *et al.*, "Zetta ζ : An efficient closed-loop embodied harness for self-evolving physical intelligence," *arXiv*, 2026.

[134] B. Luo, J. Guo, and S. Li, "Video-rsi: Recursive self-improvement of video understanding agents via harness evolution," *arXiv*, 2026.

[135] Z. Wei, F. Tan, and J. Wang, "Scale and selection: What makes automatic harness evolution work for visual-interface robot agents," *arXiv*, 2026.

[136] Y. Wang, Y. Yu *et al.*, "Vevo-harness: Causally-guarded self-evolving harness for agentic image generation," *arXiv*, 2026.

[137] H. Luo, J. Qin *et al.*, "Instagent: A recursively self-improving agent operating system for heterogeneous robot collaboration," *arXiv*, 2026.

[138] Z. Feng, Z. Pan *et al.*, "Enactopt: Diagnosis-guided harness evolution for embodied agents," *arXiv*, 2026.

[139] E. Ma, L. Zhou *et al.*, "CorrectAD: A Self-Correcting Agentic System to Improve End-to-end Planning in Autonomous Driving," in *AAAI*, 2026.

[140] R. Shao, R. Gao *et al.*, "HATS: Hardness-Aware Trajectory Synthesis for GUI Agents," in *CVPR*, 2026.

[141] W. Xiao, H. Lin *et al.*, "Self-improving vision-language-action models with data generation via residual rl," in *ICLR*, 2026.

[142] T. Xue, C. Peng *et al.*, "EvoCUA: Evolving Computer Use Agents via Learning from Scalable Synthetic Experience," in *NeurIPS*, 2026.

[143] T. Xue, Z. Liao *et al.*, "Autonomous Continual Learning for Environment Adaptation of Computer-Use Agents," in *NeurIPS*, 2026.

[144] Z. Li, H. Zhang *et al.*, "Sees: A self-evolving embodied system via failure-guided vla policy adaptation," *arXiv*, 2026.

[145] U. B. Karli and T. Fitzgerald, "Recall: Recovery experience collection for active lifelong learning in vision-language-action models," *arXiv*, 2026.

[146] X. Liu, S. Cui *et al.*, "Scene2demo: Self-evolving embodied data generation via object-action graph," in *The 2nd Workshop on Foundation Models Meet Embodied Agents at CVPR 2026*, 2026.

[147] S. Chen, Z. Xing *et al.*, "GenEvolve: Self-Evolving Image Generation Agents via Tool-Orchestrated Visual Experience Distillation," in *NeurIPS*, 2026.

[148] Y. Fang, Z. Li, H. Tong, P. Liu, and G. Chalvatzaki, "Find something you can't do: Agentic real-world reinforcement learning for self-improving vla models," *arXiv*, 2026.

[149] H. Kang, X. Ye *et al.*, "Simworld studio: Automatic environment generation with evolving coding agent for embodied agent learning," *arXiv*, 2026.

[150] C. Luo, Z. Zeng, M. Jia, Y. Du, and C. Sun, "Self-improving loops for visual robotic planning," in *ICLR*, 2026.

[151] X. Wang, W. Dai *et al.*, "GUI0: Self-Evolving Foundational GUI Agents in Super App Ecosystems," in *ACL*, 2026.

[152] F. Zhu, Z. Yan, Z. Hong, Q. Shou, X. Ma, and S. Guo, "Wmpo: World model-based policy optimization for vision-language-action models," in *ICLR*, 2026.

[153] W. Tian, Z. Zhang, Z. Liu, F. Liu, Q. Liu, and L. Wang, "Physevo: Astra can act, let it," *arXiv*, 2026.

[154] Y. Guo, T. Lee, L. X. Shi, J. Chen, P. Liang, and C. Finn, "VLAW: Iterative Co-Improvement of Vision-Language-Action Policy and World Model," in *ICML*, 2026.

[155] Z. Jiang, S. Zhou *et al.*, "Wovr: World models as reliable simulators for post-training vla policies with rl," *arXiv*, 2026.

[156] W. Chen, B. Miao *et al.*, "Learning to Adapt: Self-Improving Web Agent via Cognitive-Aware Exploration," in *CVPR*, 2026.

[157] W. Yang, Y. Zhao, B. Zhang, Y. Liu, and Q. Ye, "Self-evolving visual agent via margin reward," *arXiv*, 2026.

[158] J. Xie, S. Ruan *et al.*, "Riwanav: Recursive world-action models with self-improvement for urban navigation," *arXiv*, 2026.

[159] F. Jiang, S. Chen *et al.*, "Grounded-dreamer: Robot learning from world model synthetic data with physical grounding," *arXiv*, 2026.

[160] P. Song, Z. Liang, K. Fu, M. Wang, J. Yang, and S. Liu, "Embodiedrsi: Active continual robot learning through hypothesis-guided co-evolution," *arXiv*, 2026.

[161] Z. Bai, S. Li, T. Huang, and B. F. Karlsson, "Practice: From experience to expertise in self-evolving embodied agents," *arXiv*, 2026.

[162] L. Bärmann, J. Plewnia, A. Waibel, and T. Asfour, "Learning to forget–hierarchical episodic memory for lifelong robot deployment," *arXiv*, 2026.

[163] X. Sui, W. Zhao *et al.*, "Learning to learn from multimodal experience," *arXiv*, 2026.

[164] Y. Xu, B. Ye *et al.*, "Videoevolve: Co-evolving memory and retrieval for long video understanding," *arXiv*, 2026.

[165] T. Zou, Y. He, T. Qiu, Y. Lin, and H. Li, "Task-focused memorization for multimodal agents," *arXiv*, 2026.

[166] W. Xu, Y. Liu *et al.*, "Videoevoharness: Self evolving agent harness from unlabeled videos for video understanding," *arXiv*, 2026.

[167] H. Gao, H. Zhang, Y. Chen, and L. Ma, "Recurir: Learning when and how to revise restoration policies via recursive self-improvement," *arXiv*, 2026.

[168] T. Yu, Z. Ou *et al.*, "Coda: Co-evolution of data engines and action policies," *arXiv*, 2026.

[169] W. Tang, R. Wang *et al.*, "The trinity: Capability-driven recursive self-improvement for multimodal agents," *arXiv*, 2026.

[170] Y. Liu, H. Wang, J. Xiao, K. Zhu, X. Fu, and Z.-J. Zha, "Rsigen: Reciprocal self-improvement for agentic visual generation," *arXiv*, 2026.

[171] P. Wang, Y. Chang *et al.*, "Adarsi: On-demand recursive self-improvement for embodied agents," *arXiv*, 2026.

[172] X. Wang, W. Wu *et al.*, "Harnesspai: An evolving harness for physical ai," *arXiv*, 2026.

[173] Y. Jian, Z. Wang, Y. Hou, and H. Fan, "Learning to interact and direct: Co-evolving video director agents," *arXiv*, 2026.

[174] J. Zhang, P. Zhang *et al.*, "CineEvolve: An Agentic Self-Evolution Framework for Cinematic Multi-Shot Video Generation," *arXiv*, 2026.

[175] Y. Zhang, P. Du *et al.*, "Rewardharness: Self-evolving agentic post-training," *arXiv*, 2026.

[176] Z. Li, Z. Cao *et al.*, "Magicgui-rms: A multi-agent reward model system for self-evolving gui agents via automated feedback reflux," *arXiv*, 2026.

[177] B. Jiang, G. Zhang *et al.*, "Skillrubric: Co-evolving actor guidance and evaluator rubrics for multimodal agents," *arXiv*, 2026.

[178] Y. Zhao, H. Zhu, T. Jiang, S. Li, X. Xu, and H. H. Wang, "Co-EPG: A Framework for Co-Evolution of Planning and Grounding in Autonomous GUI Agents," in *AAAI*, 2026.

[179] Y. Lin, L. Wang *et al.*, "Jarvisevo: Towards a self-evolving photo editing agent with synergistic editor-evaluator optimization," in *CVPR*, 2026.

[180] M. Zeng, G. Tan *et al.*, "Visa: Agentic self-evolving data synthesis for multimodal instruction following," *arXiv*, 2026.

[181] Z. Wang, Y. Lu *et al.*, "Vericua: Learning proactive verification for reliable gui agent rewards," *arXiv*, 2026.

[182] L. Zhang, T. Tang *et al.*, "AtomicVLA: Unlocking the Potential of Atomic Skill Learning in Robots," in *CVPR*, 2026.

[183] Y. Liao, P. Zhou *et al.*, "Genie envisioner: A unified world foundation platform for robotic manipulation," in *ICLR*, 2026.

[184] Y. Wang, F. Zhang, D.-C. Zhan, L. Zhao, K. Wang, and J. Bian, "Co-Evolving Latent Action World Models," in *ICML*, 2026.

[185] Z. Deng, T. Ju, Z. Wu, Z. Zhang, and G. Liu, "Training High-Level Schedulers with Execution-Feedback Reinforcement Learning for Long-Horizon GUI Automation," in *CVPR*, 2026.

[186] H. Du, L. Yan, C. Flores, and A. Kadav, "Designer-rsi: Evolving procedural memory from user traffic for agentic graphic design," *arXiv*, 2026.

[187] H. Zhong, F. Faisal *et al.*, "Actionengine: From reactive to programmatic web agents via state machine memory," *arXiv*, 2026.

[188] X. Li, Y.-L. Li, Y. Wang, H. Wang, and S. Wang, "TCoT: Trajectory chain-of-thoughts for robotic manipulation with failure recovery in vision-language-action model," in *AAAI*, 2026.

[189] E. Jin, J. Hsu, Y. Xu, W. Liu, N. Haber, and J. Wu, "APIVOT: Adaptive Planning with Interleaved Vision-Language Thoughts," in *NeurIPS*, 2026.

[190] Z. Lu, H. Ren *et al.*, "Webgen-agent: Enhancing interactive website generation with multi-level feedback and step-level reinforcement learning," in *ICLR*, 2026.

[191] Z. Ke, J. Chen, J. Zhu, K. Xia, B. Zhao, and J. Gu, "REACH: Explicit Recovery Behavior for Diffusion Policies," in *CVPR*, 2026.

[192] Y. Zhang, D. Liu *et al.*, "Imagining recovery: Inference-time counterfactual realignment for vision-language-action models," *arXiv*, 2026.

[193] X. Li, X. Li, J. Gao, R. Pi, S. Hu, and W. Zhang, "Look Less, Reason More: Rollout-Guided Adaptive Pixel-Space Reasoning," in *ACL*, 2026.

[194] A.-L. Wang, G. Tang *et al.*, "Reason with Thumbnails, Answer with Focus: An Efficient and Effective Paradigm for Multimodal Grounded Visual Reasoning," in *ICML*, 2026.

[195] Y. Wang, Q. Chen *et al.*, "Simple-VGC: Enhancing Visual Grounding in Multimodal Reasoning via Adaptive Tool Composition," in *ACL*, 2026.

[196] K. Mei, J. Guo *et al.*, "R-WoM: Retrieval-augmented World Model For Computer-use Agents," in *ICLR*, 2026.

[197] Z. Ning, X. Gao *et al.*, "V-ABS: Action-Observer Driven Beam Search for Dynamic Visual Reasoning," in *ICML*, 2026.

[198] X. Zhang, X. Zhang *et al.*, "Generative universal verifier as multimodal meta-reasoner," in *ICLR*, 2026.

[199] Z. Chen, J. Chen, S. Arik, M. Sra, T. Pfister, and J. Yoon, "CoDA: Agentic systems for collaborative data visualization," in *ICLR*, 2026.

[200] P. Chen, S. Chen, Q. Ye, X. Chen, X. Li, and W. Xiang, "MaDS: Long-Horizon GUI Automation via Synergizing Dual-Layer Memory and Multi-Round Debate," in *ACL*, 2026.

[201] M. Andrade, J. Cha, B. Ho, V. Srihari, K. Yadav, and Z. Kira, "Let's Think in Two Steps: Mitigating Agreement Bias in MLLMs with Self-Grounded Verification," in *ICLR*, 2026.

[202] Q. Sun, M. Li *et al.*, "OS-Sentinel: Towards Safety-Enhanced Mobile GUI Agents via Hybrid Validation in Realistic Workflows," in *ACL*, 2026.

[203] Y. Guo, Y. Hou, W. Ma, M. Tang, and M.-H. Yang, "Pursuing minimal sufficiency in spatial reasoning," in *ICLR*, 2026.

[204] Z. Huang, Y. Ji *et al.*, "Learning to Select Visual Tools from Experience," in *CVPR*, 2026.

[205] J. Levy, T. Westenbroek *et al.*, "Simulation distillation: Pretraining world models in simulation for rapid real-world adaptation," *arXiv*, 2026.

[206] H. Zhou, J. Watson, A. Lei, and I. Posner, "Benchmarking world models for continual learning on compositional tasks," *arXiv*, 2026.

[207] Z. Liu, L. Huang *et al.*, "Skilllens: Visual skill cards for retrieval-augmented gui action prediction and on-policy distillation," *arXiv*, 2026.

[208] Z. Fan, H. Yu *et al.*, "Skill-guided continuation distillation for gui agents," *arXiv*, 2026.

[209] C. Li, J. Liu *et al.*, "Experience Transfer for Multimodal LLM Agents in Minecraft Game," in *CVPR*, 2026.

[210] W. Jiang, Z. Cai *et al.*, "Manimagent: Self-evolving multimodal agents for visual education," *arXiv*, 2026.

[211] T. Kwon, D. Choi *et al.*, "Embodied Agents Meet Personalization: Investigating Challenges and Solutions Through the Lens of Memory Utilization," in *ICLR*, 2026.

[212] L. Che, B. Liu *et al.*, "Lvmh: A meta-optimization harness for long-video agent memory," *arXiv*, 2026.

[213] B. Chen, B. Zhang *et al.*, "Reflect, revise, reuse: Training-free skill evolution for gui agents," *arXiv*, 2026.

[214] Z. Jiang, L. An *et al.*, "Visualskill: Multimodal skills for computer-use agents," *arXiv*, 2026.

[215] J. Zhou, S. Lin, J. Li, S. Fu, G. Zhou, and Q. Wu, "Automating the design of embodied agent architectures," *arXiv*, 2026.

[216] Z. Chen, Y. Sun *et al.*, "KPHarness: Co-Evolving Knowledge and Probing for Long-Horizon Gaming Harnesses," *arXiv*, 2026.

[217] Y. Jiang, C. Xu, F. Gao, and L. Zhao, "Harnessminusminus-v: Recursive self-improvement for visual agent harness token efficiency," *XSci preprint*, 2026.

[218] Z. Peng, W. Ding *et al.*, "Counterfactual VLA: Self-Reflective Vision-Language-Action Model with Adaptive Reasoning," in *CVPR*, 2026.

[219] H. Liu, C. Kim, B. Liu, M. Liu, and Y. Zhu, "Pretrained Vision-Language-Action Models are Surprisingly Resistant to Forgetting in Continual Learning," in *ICML*, 2026.

[220] J. Zhu, Y. Hong *et al.*, "Can vision-language-action models learn from real-world data continually without forgetting," *arXiv*, 2026.

[221] S. Raghavan, Z. Yuan, F. Zhu, and B.-C. Min, "Does continual imitation learning remain grounded? a language-perturbed benchmark for robotic task retention," *arXiv*, 2026.

[222] L. Guo, W. Liu, Y. W. Heng, T.-H. P. Chen, and Y. Wang, "Agent-sama: state-aware mobile assistant," in *AAAI*, 2026.

[223] L. Sun, J. Zhang, S. Wang, and Z. Wei, "MAGNET: Towards Adaptive GUI Agents with Memory-Driven Knowledge Evolution," in *ACL*, 2026.

[224] D. Ye, D. Zhao *et al.*, "Videoresearcher: Self-improving tool design for long-video understanding," *arXiv*, 2026.

[225] R. Yang, H. Chen *et al.*, "EmbodiedBench: Comprehensive Benchmarking Multi-modal Large Language Models for Vision-Driven Embodied Agents," in *ICML*, 2025.

[226] L. Hu, X. Liang, and L. Zhu, "From interaction traces to persistent skills: Online evolution for computer-use agents," *arXiv*, 2026.

[227] X. Li, X. Gu *et al.*, "Reflect after failure: Brain-inspired self-evolving agents for embodied spatial intelligence," *arXiv*, 2026.

[228] J. Wang, Y. Li *et al.*, "Visualevolve: A skill self-evolution framework for visual reasoning," *arXiv*, 2026.

[229] X. Xiao, J. Zhong *et al.*, "Geoforge: Non-parametric self-evolving agents for earth-observation reasoning," *arXiv*, 2026.

[230] K. Wang, H. Jie, Y. Yan, Z. Heng, and Z. Li, "Regenharness: A robot agent harness with evidence-gated recursive self-improvement," *arXiv*, 2026.

[231] S. Karten, J. Zhang *et al.*, "Continual harness: Online adaptation for self-improving foundation agents," *arXiv*, 2026.

[232] T. Du, H. Guo *et al.*, "From VLM Planners to Robotic Agents: Self-Evolving Harnesses for History-Dependent Manipulation," *arXiv*, 2026.
[233] B. Liu, Y. Zhu *et al.*, "LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning," in *NeurIPS*, 2023.
[234] Z. Wang, C. Chen *et al.*, "ActiView: Evaluating Active Perception Ability for Multimodal Large Language Models," in *ACL*, 2025.
[235] A. Gupta, S. Konan, T. Yin, G. Sarch, and Z. Liu, "Agentvqa: A unified benchmark for agentic visual understanding," *arXiv*, 2026.
[236] C. Fu, H. Yuan *et al.*, "Video-mme-v2: Towards the next stage in benchmarks for comprehensive video understanding," *arXiv*, 2026.
[237] Q. Wang, W. Huang *et al.*, "ENACT: Evaluating Embodied Cognition with World Modeling of Egocentric Interaction," in *ICLR*, 2026.
[238] R. Chen, Q. Li *et al.*, "MPR-GUI: Benchmarking and Enhancing Multilingual Perception and Reasoning in GUI Agents," in *ACL*, 2026.
[239] Y. Li, E. Zhou *et al.*, "Activearena: Benchmarking and understanding active perception in robotic manipulation," *arXiv*, 2026.
[240] A. Majumdar, A. Ajay *et al.*, "OpenEQA: Embodied Question Answering in the Era of Foundation Models," in *CVPR*, 2024.
[241] M. Yuan, Z. Zhou *et al.*, "Osworld2.0: Benchmarking computer use agents on long-horizon real-world tasks," *arXiv*, 2026.
[242] I. Kerboua, F. P. Zadeh *et al.*, "WebArena-Pro: A Heterogeneous, Multimodal, Reproducible Benchmark for Web Agents," in *NeurIPS*, 2026.
[243] S. Yu, L. Shu *et al.*, "Ego2Web: A Web Agent Benchmark Grounded in Egocentric Videos," in *CVPR*, 2026.
[244] Z. Zheng, T. Cui *et al.*, "NaturalGAIA: A Verifiable Benchmark and Hierarchical Framework for Long-Horizon GUI Tasks," in *ACL*, 2026.
[245] J. Ai, Y. Feng *et al.*, "ProSoftArena: Benchmarking Hierarchical Capabilities of Multi-modal Agents in Professional Software Environments," in *CVPR*, 2026.
[246] J. Y. Koh, R. Lo *et al.*, "VisualWebArena: Evaluating Multimodal Agents on Realistic Visual Web Tasks," in *ACL*, 2024.
[247] Y. Liu, Z. Lu *et al.*, "Memu-bench: Benchmarking gui agents on long-horizon user interactions," *arXiv*, 2026.
[248] A. Tupini, L. Liden, R. Tan, Y. Wang, and J. Gao, "Asgardbench – evaluating visually grounded interactive planning under minimal feedback," *arXiv*, 2026.
[249] X. Xu, P. Bu *et al.*, "DeepPHY: Benchmarking agentic VLMs on physical reasoning," in *AAAI*, 2026.
[250] S. Karten, J. Grigsby *et al.*, "The pokeagent challenge: Competitive and long-context learning at scale," *arXiv*, 2026.
[251] M. Shridhar, J. Thomason *et al.*, "ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks," in *CVPR*, 2020.
[252] H. Jia, J. Liao *et al.*, "OSWorld-MCP: Benchmarking MCP Tool Invocation In Computer-Use Agents," in *ICLR*, 2026.
[253] Z. Su, J. Gao *et al.*, "AgentVista: Evaluating Multimodal Agent in Ultra-Challenging Realistic Visual Scenarios," in *NeurIPS*, 2026.
[254] Y. Liu, T. Niu *et al.*, "Egobench: An interactive egocentric multimodal benchmark for tool-using agents," *arXiv*, 2026.
[255] Y. Gao, L. Shu *et al.*, "3DCodeBench: Benchmarking Agentic Procedural 3D Modeling Via Code," in *NeurIPS*, 2026.
[256] L. Liu, B. Gong, A. D. Parakkat, and M. Ovsjanikov, "3dharnessbench: Probing agentic 3d-to-code capabilities of frontier vision-language models," *arXiv*, 2026.
[257] A. Gandhi, V. Suryanarayanan *et al.*, "PPT-Eval: A Benchmark for Computer-Use Agents on PowerPoint Tasks," in *ICML*, 2026.
[258] J. He, C. Fu *et al.*, "From evaluation to evolution: Benchmarking image editing and evolving image models," *arXiv*, 2026.
[259] Y. Wang, R. Fang *et al.*, "Refcraft: Benchmarking visual-reference-conditioned agentic generation through web development," *arXiv*, 2026.
[260] X. Xue, Z. Lu, D. Huang, Z. Wang, W. Ouyang, and L. Bai, "ComfyBench: Benchmarking LLM-based Agents in ComfyUI for Autonomously Designing Collaborative AI Systems," in *CVPR*, 2025.
[261] X. Liu, T. Zhang *et al.*, "VisualAgentBench: Towards Large Multimodal Models as Visual Foundation Agents," in *ICLR*, 2025.
[262] T. Cao, B. Lim *et al.*, "VPI-Bench: Visual Prompt Injection Attacks for Computer-Use Agents," in *ICLR*, 2026.
[263] X. Hu, H. Li *et al.*, "CUA-Recovery: Benchmarking Error Recovery in Long-Horizon Computer-Use Agents," *arXiv*, 2026.
[264] L. Yang, Z. Wang *et al.*, "Probench: Benchmarking gui agents with accurate process information," in *AAAI*, 2026.
[265] Z. Navasardyan, T. Danielyan, and H. Davtyan, "Failbench: How reliable are vlms at judging robot task success?" *arXiv*, 2026.
[266] Z. Ying, L. Wang *et al.*, "AGENTSAFE: Benchmarking the Safety of Embodied Agents on Hazardous Instructions," in *CVPR*, 2026.
[267] X. Lu, Z. Chen *et al.*, "Is-bench: Evaluating interactive safety of vlm-driven embodied agents in daily household tasks," in *AAAI*, 2026.
[268] Z. Tang, G. Zhao *et al.*, "RecoverBench: A Systematic Benchmark for Error Recovery in Robotic Manipulation," in *NeurIPS*, 2026.
[269] B. Zhang, Y. Zhang *et al.*, "SafeVLA: Towards Safety Alignment of Vision-Language-Action Model via Constrained Learning," in *NeurIPS*, 2025.
[270] S. Powers, E. Xing, E. Kolve, R. Mottaghi, and A. Gupta, "CORA: Benchmarks, Baselines, and Metrics as a Platform for Continual Reinforcement Learning Agents," in *Proceedings of The 1st Conference on Lifelong Learning Agents*, 2022.
[271] G. Liu, P. Zhao *et al.*, "Memgui-bench: Benchmarking memory of mobile gui agents in dynamic environments," *arXiv*, 2026.
[272] L. Long, Y. He *et al.*, "Seeing, listening, remembering, and reasoning: A multimodal agent with long-term memory," in *ICLR*, 2026.
[273] Y. Yang, G. Sun, Y. Li, W. Li, Z. Ma, and C. Zhang, "SLVMBench: Skill Learning from Video Memory," in *NeurIPS*, 2026.
[274] R. Forte, G. Lando, and A. Furnari, "EgoStream: A Diagnostic Benchmark for Streaming Episodic Memory in Egocentric Vision," in *NeurIPS*, 2026.
[275] C. Liu, Y. YANG *et al.*, "WorldMemArena: Evaluating Multimodal Agent Memory Through Action–World Interaction," in *NeurIPS*, 2026.
[276] Y. Bei, T. Wei *et al.*, "Mem-gallery: Benchmarking multimodal long-term conversational memory for mllm agents," in *ACL*, 2026.
[277] K. Wang and C. Zhang, "MemLeak: Diagnosing Information Leaks in Multimodal Agent Memory," in *NeurIPS*, 2026.
[278] S. Wang, B. Liu *et al.*, "Explore with Long-term Memory: A Benchmark and Multimodal LLM-based Reinforcement Learning Framework for Embodied Exploration," in *CVPR*, 2026.
[279] X. Wang, G. Zhang, H. Xu, Z. Zhang, T. Wang, and L. Fan, "Evonav-bench: Benchmarking lifelong navigation in evolving environments," *arXiv*, 2026.
[280] J. Krantz, S. Banerjee *et al.*, "Iterative vision-and-language navigation," in *CVPR*, 2023.
[281] J. Huang, Y. Chen *et al.*, "Embodiedclbench: Evaluating continual learning for self-evolving embodied agents," *arXiv*, 2026.
[282] X. Wang, B. Li *et al.*, "Sear: Self-evolving agentic robotics," *arXiv*, 2026.
[283] Z. Li, J. Yu *et al.*, "Where is physical rsi capability from? an observational scaling law of llm-as-policy," *arXiv*, 2026.
[284] OpenRSI Foundation, "OpenRSI Index," https://github.com/OpenRSI-Foundation/OpenRSI-Index, 2026.
[285] B. Xu, J. Yang *et al.*, "Selfimprove-bench: Evaluating generalization in self-improving agents," *arXiv*, 2026.
[286] T. E. Lee, N. R. Ke *et al.*, "CURATE: Automatic curriculum learning for reinforcement learning agents through competence-based curriculum policy search in structured task spaces," *TMLR*, 2026.
[287] J. Zhang, J. Wang *et al.*, "Flex-rsi: Benchmarking recursive self-improvement of frontier agents," *arXiv*, 2026.
[288] Y. Chi, W. Li *et al.*, "Ai4ai-bench: Benchmarking llm agents in algorithmic design for recursive self-improvement," *arXiv*, 2026.
[289] RSI-Exam Team, "Rsi-exam: Benchmarking recursive self-improvement through executable research," https://github.com/aiming-lab/RSI-Exam, 2026.
[290] A. Ramesh, S. Devic *et al.*, "AutoResearchExam," 2026. [Online]. Available: https://benchmarks.bespokelabs.ai/autoresearchexam/
[291] O. Chen, G. Jaff, L. Nashold, and R. Krishnan, "Vals rsi index," 2026. [Online]. Available: https://vals-ai.com/benchmarks/rsi_index
[292] H. Wu, D. Li, B. Chen, and J. Li, "LongVideoBench: A Benchmark for Long-context Interleaved Video-Language Understanding," in *NeurIPS*, 2024.
[293] Y. Zhao, H. Zhang *et al.*, "MMVU: Measuring Expert-Level Multi-Discipline Video Understanding," in *CVPR*, 2025.
[294] J. Cheng, Y. Ge, T. Wang, Y. Ge, J. Liao, and Y. Shan, "Video-Holmes: Can MLLM Think Like Holmes for Complex Video Reasoning?" in *ECCV*, 2026.
[295] H. Wang, W. Feng *et al.*, "Search beyond what can be taught: Evolving the knowledge boundary in agentic visual generation," *arXiv*, 2026.
[296] S. Nasiriany, S. Nasiriany, A. Maddukuri, and Y. Zhu, "RoboCasa365: A large-scale simulation framework for training and benchmarking generalist robots," in *ICLR*, 2026.
[297] Q. Meng, Y. Wang *et al.*, "Agent harness for large language model agents: A survey," *Preprints*, 2026.