

INFORMATION SCIENCE

Self-evolving Embodied AI

Tongtong Feng, Xin Wang* and Wenwu Zhu*

ABSTRACT

Embodied Artificial Intelligence (AI) is an intelligent system formed by agents and their environment through active perception, embodied cognition, and action interaction. Existing embodied AI remains confined to human-crafted setting, in which agents are trained on given memories and construct models for given tasks, enabling fixed physical embodiments to interact with relatively static environments. Such methods fail in in-the-wild setting characterized by variable physical embodiments and dynamic open environments. This paper introduces self-evolving embodied AI, a new paradigm in which agents operate based on their changing physical embodiment and environment through five software-level components, namely memory self-updating, task self-switching, environment self-prediction, embodiment self-adaptation, and model self-evolution, together with a hardware-level component for physical embodiment evolution, aiming to achieve continually adaptive intelligence through software-hardware co-evolution. Specifically, we present the definition, framework, components, mechanisms, roadmap, and evaluation indicators of self-evolving embodied AI, systematically review state-of-the-art works for realized components, discuss practical applications, and point out future research directions. We believe that self-evolving embodied AI enables agents to autonomously learn and interact with environments in a human-like manner and provides a new perspective toward general artificial intelligence.

Keywords: Embodied AI, Self-evolving, Autonomous

1. Introduction

Embodied Artificial Intelligence (AI) originates from the Embodied Turing Test by Alan Turing in 1950 [1], which emphasizes that intelligence should be evaluated through an agent's ability to perceive and act in the physical world in addition to abstract symbol computation. Specifically, embodied AI refers to intelligent systems [2] formed by agents and their environment through active perception, embodied cognition, and action interaction. Unlike disembodied AI that operates on static data or abstract symbols, embodied AI emphasizes the closed-loop coupling between perception, cognition, and action [3], enabling agents to interact with physical environments in a grounded manner. As such, embodied AI benefits in promising potential in a wide range of applications, including robotics [4], autonomous driving [5], and other real-world interactive systems [6] etc.

Despite its rapid progress, existing embodied AI remains confined to human-crafted setting [7, 8], in which agents are trained with given mem-

ories and construct models for given tasks, thus enabling fixed physical embodiments to interact with relatively static environments [9]. The existing embodied AI learning paradigm [10] is optimized for given tasks under predefined objectives, and the performance relies heavily on external human guidance and empirical specification [11], including human-given tasks, pre-designed models, manually-collected datasets, fixed-configuration physical embodiments, and specified experimental environments.

However, real-world scenarios often involve in-the-wild setting. On the one hand, in-the-wild environments are often dynamic and open [12,13], with continually changing objects, dynamics, and interaction patterns that cannot be fully predefined or exhaustively enumerated during training. On the other hand, in-the-wild deployment often requires variable physical embodiments [14], where agents differ in morphology, sensing, actuation, computational configurations, and physical constraints, resulting in models learned under fixed physical embodiments being difficult to adapt. Unfortunately, ex-

Department of
Computer Science and
Technology, Beijing
National Research
Center for Information
Science and
Technology, Tsinghua
University, Beijing
100084, China

*Corresponding authors.

Email:
wwzhu@tsinghua.edu.cn;
xin.wang@tsinghua.edu.cn.

Received: XX XX Year;

Revised: XX XX Year;

Accepted: XX XX Year

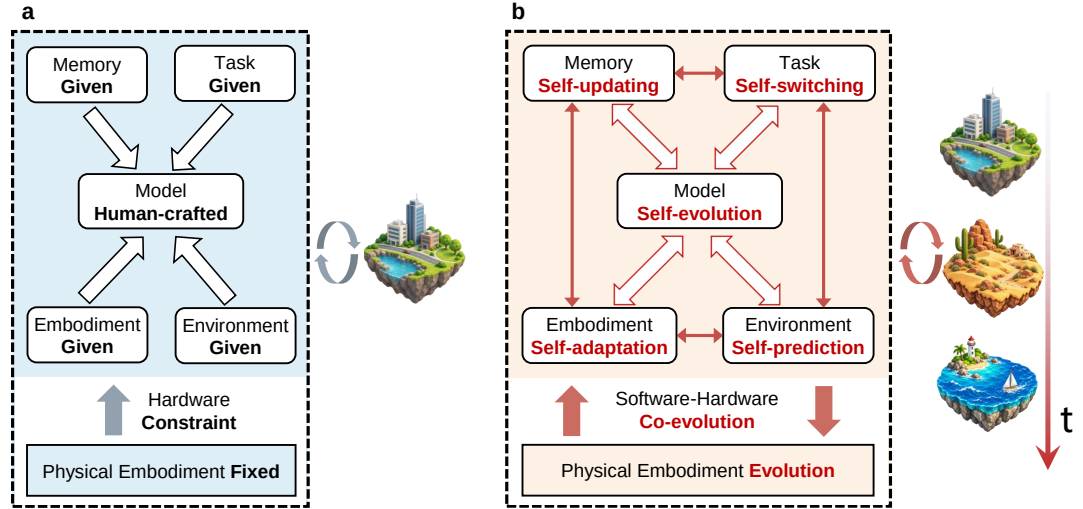


Figure 1. Comparison between existing embodied AI and self-evolving embodied AI. (a) Existing embodied AI operates with given memory, given task, given embodiment representation, and given environment, using a human-crafted model, while its fixed physical embodiment imposes hardware constraints. This paradigm relies heavily on external human guidance and empirical configurations. (b) Self-evolving embodied AI operates based on its changing physical embodiment and environment through five software-level components, namely memory self-updating, task self-switching, environment self-prediction, embodiment self-adaptation, and model self-evolution, together with a hardware-level component for physical embodiment evolution. Here, embodiment self-adaptation updates the software-level representation of the physical embodiment, whereas physical embodiment evolution changes the hardware-level embodiment itself. In this paradigm, agents can achieve continually adaptive intelligence through software-hardware co-evolution.

isting embodied AI fails in in-the-wild setting.

Motivated by neuroscience [15] and cognitive science [16], we introduce a new paradigm of 'self-evolving embodied AI', as illustrated in Fig. 1. Self-evolving embodied AI requires agents to operate based on their changing physical embodiment and environment through five software-level components, namely memory self-updating, task self-switching, environment self-prediction, embodiment self-adaptation, and model self-evolution, together with a hardware-level component for physical embodiment evolution. Here, embodiment self-adaptation updates the software-level representation of the physical embodiment, whereas physical embodiment evolution changes the hardware-level embodiment itself. Software evolution guides physical embodiment evolution, which in turn reshapes software evolution. In this paradigm, agents can achieve continually adaptive intelligence through software-hardware co-evolution, enabling embodied AI to move from human-crafted setting to in-the-wild setting.

Specifically, we first present the definition, framework, components, mechanisms, roadmap, and evaluation indicators of self-evolving embodied AI. We then systematically review state-of-the-art works for each self-evolving component. Furthermore, we discuss practical applications that demonstrate the advantages of self-evolving embodied AI in variable physical em-

bodiments and dynamic open environments. Finally, we summarize future research directions toward building controllable, trustworthy, and swarm self-evolving embodied AI.

It should be noted that self-evolving embodied AI is currently a vision-guided paradigm rather than a review of a mature technical system. No existing system has yet realized full-stack autonomous co-evolution between all five software-level components and a hardware-level component. Existing studies should be regarded as partial realizations that provide preliminary building blocks toward the ideal full-loop mechanism. Nevertheless, we anticipate with full confidence that self-evolving embodied AI is able to learn autonomously in a human-like manner and provide a new perspective toward general artificial intelligence.

2. Self-evolving Embodied AI

This section introduces 'self-evolving embodied AI', a new paradigm that aims to move embodied AI beyond human-crafted setting toward in-the-wild setting. We first provide a precise definition and unified framework and then elaborate on its five software-level components and hardware-level component. We next explain two coupled self-evolving mechanisms, namely co-evolution among software-level components and software-hardware co-evolution. Building on these mech-

Table 1. Comparison between adaptive intelligence and self-evolving intelligence.

Type	Level of Change	Object of Change	Goal of Change
Adaptive	Behavior-level adjustment within a given system	Model parameters	Adaptive capabilities
Self-evolving	System-level self-correction and self-improvement	Memory organization, task formulation, world modeling, embodiment representation, model architecture, and physical embodiment itself	Improved capabilities, structures, and mechanisms

anisms, we formulate a closed-loop roadmap for component triggering, software-hardware co-evolution, constraint verification, and continual state transition. Finally, we introduce evaluation indicators for measuring component-level and system-level self-evolution.

2.1 The Definition and Framework

This subsection establishes the conceptual foundation of self-evolving embodied AI. We provide a precise definition of self-evolving embodied AI and then introduce a unified framework to describe how self-evolution is organized.

Definition 1 (Self-evolving embodied AI). An agent operates based on its changing physical embodiment and external environment through five software-level components, namely memory self-updating, task self-switching, environment self-prediction, embodiment self-adaptation, and model self-evolution, together with a hardware-level component for physical embodiment evolution, all of which are coupled through software-hardware co-evolution to achieve continually adaptive intelligence.

As shown in Table 1, adaptive intelligence [17] and self-evolving intelligence represent two different intelligence types in embodied scenarios. Adaptive intelligence focuses on post-deployment behavioral adjustment under changing conditions, such as shifts in data distributions, task requirements, or environmental states. It changes the agent's behavior within a given agent system. Self-evolving intelligence, however, refers to feedback-driven self-correction and self-improvement of the agent system itself. The key distinction is that adaptive intelligence modifies the behavior produced by a given system, whereas self-evolving intelligence modifies the system that produces behavior. In self-evolving embodied AI, the object of change includes memory organization, task formulation, world modeling, embodiment representation, model architecture, and the physical embodiment itself.

Framework. Figure 1b illustrates the framework of self-evolving embodied AI. Specifically, the software layer is organized around five tightly coupled modules, namely memory self-updating, task self-switching, environment self-prediction, embodiment self-adaptation, and model self-evolution. At the hardware level, physical embodiment evolution enables changes to the agent's physical embodiment itself. The five software-level modules are not independent, but evolve jointly through continuous interaction with the environment and with each other. Updates in one module may induce adjustments in others. This organization is reflected by bidirectional information exchange among modules across time. For example, changes in physical embodiment constraints may alter the relevance of stored memory, which in turn affects task switching and model evolution. Software evolution can also guide physical embodiment evolution, which in turn reshapes software evolution. Together, the software-level components and the hardware-level component form a unified evolutionary loop that supports continually adaptive intelligence under variable physical embodiments and dynamic open environments.

For comparison, Fig. 1a summarizes the framework of existing embodied AI [2,4,5,7,14, 18]. Specifically, the agent with given physical embodiment, memory, task, uses a human-crafted model for predefined objectives in a specific environment. This framework is largely unidirectional, shown as one-way arrows pointing toward the human-crafted model.

The contrast between the two frameworks highlights a fundamental difference in how embodied AI is realized. Existing embodied AI relies on given components and human-crafted models with external human guidance and empirical configurations, whereas self-evolving embodied AI organizes memory self-updating, task self-switching, environment self-prediction, embodiment self-adaptation, and model self-evolution as five software-level components, while these components co-evolve with

a hardware-level component for physical embodiment evolution. Compared to existing embodied AI operating under relatively static environments, by replacing unidirectional optimization with bidirectional coupling among self-evolving modules and between the software and hardware levels, self-evolving embodied AI enables continual adaptation with autonomous evolution in dynamic open environments.

Recent studies have shown a growing interest in self-evolving agents [19–24], which aim to improve adaptability through continual interaction with the environment. Specifically, those methods explore self-evolution by using prompt engineering [25,26], memory selection [27–30], and tool refinement technologies [31], and are often built upon pretrained Large Language Models (LLMs) as unified end-to-end backbones [32,33]. Those methods demonstrate the potential of self-evolution without explicit human intervention, but environment perception blocks and physical embodiment configurations are typically treated as fixed once deployed [34,35], limiting their ability to support long-term autonomy under variable physical embodiments and dynamic environments.

In summary, to the best of our knowledge, this paper is the first systematic formalization of self-evolution as a top-level paradigm for embodied AI. Specifically, self-evolving embodied AI arises from closed-loop evolution among five software-level components and their bidirectional co-evolution with a hardware-level component, driven by the agent’s self-state transitions by action-feedback interactions.

2.2 The Components

This subsection elaborates on the core components of self-evolving embodied AI. For each component, we explain what evolves, why evolution is necessary, when evolution is triggered, how evolution is realized.

Memory self-updating. What evolves in memory self-updating is the agent’s internal memory representation. Experience is continuously generated through the agent’s interaction with the environment, while memory self-updating determines how experience is selectively retained, revised, or discarded over time. Why evolution is necessary lies in the fact that fixed memories or static datasets cannot support long-term adaptation under environmental distribution shift and changing physical embodiments or tasks. In in-the-wild environments, storing all past experi-

ences is neither feasible nor desirable. When evolution is triggered, previously stored experiences become outdated, irrelevant, or misleading with respect to the agent’s current physical embodiment constraints, task objectives, or environmental dynamics. How evolution is realized is through selective memory updating mechanisms, such as memory self-editing, memory self-organization, and memory self-distillation, which prioritize experiences based on relevance, novelty, uncertainty, or long-term utility. This component has relatively high feasibility, as recent agent-memory studies [36–38] have already explored memory editing, organization, and distillation for long-horizon interaction. However, key bottlenecks remain in long-term memory reliability, conflict resolution among inconsistent experiences, privacy of stored interaction histories, and preventing outdated memories from misleading future decisions.

Task self-switching. What evolves in task self-switching is the agent’s internal representation of task objectives. Rather than optimizing predefined objectives of given tasks, task self-switching improves the agent’s ability to autonomously adjust what it is trying to achieve over time. Why evolution is necessary lies in the fact that predefined objectives of given tasks cannot adequately capture the changing goals, constraints, and opportunities encountered in dynamic open environments. In-the-wild setting rarely present stable, well-defined goals; over time, predefined objectives of given tasks may limit autonomy and prevent effective adaptation. When evolution is triggered, current task formulations become infeasible, sub-optimal, or misaligned with the agent’s internal state, physical embodiment constraints, or environmental dynamics. How evolution is realized is through autonomous task self-switching mechanisms, such as task self-selection and task self-generation, enabling agents to continuously adjust what they aim to achieve without explicit human re-specification. This component is scientifically reasonable [39,40], but its engineering feasibility is more challenging than memory updating. The main bottlenecks lie in goal drift, task misalignment, unsafe task generation, and the lack of stable constraints for deciding when an agent should switch, decompose, or generate tasks autonomously.

Environment self-prediction. What evolves in environment self-prediction is the agent’s internal representation of the external world. It captures

how agents continuously update their understanding of environmental dynamics and make future predictions of the external world. Why evolution is necessary lies in the fact that real-world environments are non-stationary and cannot be fully characterized by fixed or offline-learned models. Agents need to maintain and refine predictive representations that anticipate future states, rewards, and action-conditioned interaction sequences. When evolution is triggered, newly observed environment states contradict prior predictions, reveal previously unseen dynamics, or expose long-term dependencies that were not captured before. How evolution is realized is through continual refinement of world models based on ongoing interaction, such as understanding world models, generative world models, and collaborative world models, allowing agents to update their internal representation of the external world and supporting adaptive planning and decision-making over time. This component also has relatively high feasibility because world models [12,13,41,42] have shown strong potential in representation learning, imagination, and model-based planning. Nevertheless, practical bottlenecks remain in long-horizon rollout consistency, uncertainty calibration, causal prediction, and action-conditioned generalization under dynamic open environments.

Embodiment self-adaptation. What evolves in embodiment self-adaptation is the agent's internal representation of its own physical state. Self-evolving embodied AI needs to adapt to changes in the physical embodiment's heterogeneous morphology, sensing capabilities, actuation limits, computational configurations, and physical constraints. Why evolution is necessary lies in the fact that embodiments may vary across platforms or change over time due to reconfiguration, causing policies learned under fixed physical embodiments to fail. When evolution is triggered, discrepancies arise between expected and actual physical embodiment states, indicating that previous embodiment assumptions are no longer valid. How evolution is realized is through software-level embodiment-aware adaptation, such as embodiment self-transfer, embodiment self-calibration and embodiment self-recovery, enabling agents to maintain functionality across variable physical embodiments. This component is feasible as a partial function [43–46], especially in cross-embodiment transfer, on-line calibration, and damage recovery. However, reliable deployment remains difficult due to sim-

to-real gaps, and the need to ensure physical feasibility under morphology, sensing, actuation, or energy changes.

Model self-evolution. What evolves in model self-evolution is the agent's internal model design, including model architectures, optimization strategies, and evaluation criteria, rather than only model parameters. Why evolution is necessary lies in the fact that fixed architectures, given training strategies, and predefined evaluation metrics cannot remain effective under long-term changes in memory, tasks, environments, and physical embodiments. As operating conditions evolve, models that were previously well-optimized may become inefficient, misaligned, or inadequate. When evolution is triggered, persistent performance degradation, increasing uncertainty, or systematic mismatch between evaluation outcomes and real-world behavior indicates that current model designs are no longer appropriate. How evolution is realized is through adaptive modification of model architectures, adjustment of optimization strategies, and refinement of evaluation criteria, such as model self-restructuring, model self-optimization, and model self-evaluation, enabling the learning process itself to evolve and remain aligned with long-horizon autonomous operation. Compared with the other components, model self-evolution is still the most under-explored and least mature. Existing works [47–49] provide preliminary evidence for model restructuring, self-optimization, and self-evaluation, but most of them remain limited to modular routing, adapter composition, self-refinement, or LLM-based judging. Critical bottlenecks include insufficient theoretical guarantees for convergence, stability, safety, and alignment, as well as limited empirical support for autonomous model evolution in embodied closed-loop systems.

Physical embodiment evolution. Physical embodiment evolution encompasses reconfiguration within a predefined hardware architecture and design space, as well as transformation that alters or expands the hardware architecture or design space itself. Reconfiguration changes modules, sensors, or actuators while preserving the underlying architecture, whereas transformation may fundamentally change morphology, actuation mechanisms, or computing architecture, such as a transition from a fish-inspired to a mammal-inspired embodiment or from a conventional von Neumann architecture to a compute-in-memory (CIM) architec-

ture. It therefore permits previously unseen hardware architectures and embodiment paradigms to emerge. Physical embodiment evolution is necessary when a fixed physical embodiment constrains the agent's functional capabilities or adaptation to new tasks and environments. It is triggered when software-level adaptation alone cannot resolve a mismatch between the capabilities of the current physical embodiment and the requirements of present or anticipated tasks and environments. It is realized through software-hardware co-evolution, in which software guides changes to the physical embodiment, while the resulting changes reshape perception and action capabilities and thereby provide new feedback for software evolution. Current realizations mainly take the form of morphological reconfiguration [50,51] and functional reconfiguration [52], whereas transformation remains largely unrealized in current embodied AI systems because fundamentally new hardware architectures may require lengthy design, fabrication, and validation cycles.

2.3 Self-evolving Mechanisms

Self-evolving embodied AI operates through two coupled mechanisms: co-evolution among software-level components and software-hardware co-evolution.

Co-evolution among software-level components. Memory self-updating, task self-switching, environment self-prediction, embodiment self-adaptation, and model self-evolution do not evolve independently. Instead, changes in one component propagate to the others through action-feedback interactions. For example, embodiment self-adaptation directly affects the entire self-evolving loop. When the agent detects a mismatch between expected and actual physical embodiment states, such as sensor degradation, actuator failure, or morphology change, this mismatch first triggers environment self-prediction to revise body-environment dynamics. It then guides memory self-updating to prioritize experiences relevant to the current physical embodiment, constrains task self-switching by filtering infeasible objectives, and further drives model self-evolution to adjust architectures or optimization strategies.

Software-hardware co-evolution. When software-level adaptation alone cannot satisfy the agent's operating requirements, a mismatch between current hardware capabilities and task or envi-

ronmental demands may trigger physical embodiment evolution. Depending on whether the required hardware changes remain within or extend beyond the existing hardware architecture and design space, physical embodiment evolution may be realized through reconfiguration or transformation, respectively. Software and hardware subsequently co-evolve, with changes in the physical embodiment reshaping perception and action capabilities and requiring corresponding software adaptation. As shown in Fig. ??, this bidirectional interaction forms the software-hardware co-evolution loop. Repeated operation of this loop enables software updates and physical embodiment changes to remain coordinated as task and environmental requirements evolve.

Together, the two mechanisms enable continually adaptive intelligence with autonomous evolution in variable physical embodiments and dynamic open environments.

2.4 Self-evolving Roadmap

Figure 2 presents a closed-loop roadmap consisting of state representation, component triggering, software-hardware co-evolution, constraint verification, and state transition. At time t , the joint state is defined as:

$$Z_t = (S_t, H_t, E_t), \quad (1)$$

$$S_t = (M_t, T_t, B_t, N_t, \Phi_t). \quad (2)$$

where Z_t is the joint state, S_t is the software state, H_t is the hardware configuration, and E_t is the external environment. Within S_t , $M_t, T_t, B_t, N_t, \Phi_t$ denote memory, task, embodiment representation, environment representation, and model, respectively. In particular, B_t describes software adaptation to the physical embodiment, whereas H_t describes the physical hardware itself.

The trigger gates G_t select the software components and the hardware component that need to be activated. The selected software change ΔS and hardware change ΔH are jointly optimized:

$$\begin{aligned} (\Delta S_t^*, \Delta H_t^*) = \arg \max_{\Delta S, \Delta H} & \\ & [J(S_t \oplus \Delta S, H_t \oplus \Delta H; E_{t:t+k}) \\ & - \lambda_S C_S(\Delta S) - \lambda_H C_H(\Delta H)], \end{aligned} \quad (3)$$

$$\begin{aligned} \text{s.t.} \quad & R(Z_{t+1}) \leq \epsilon, \\ & C_S(\Delta S) + C_H(\Delta H) \leq \beta, \\ & H_t \oplus \Delta H \in \mathcal{H}_{\text{safe}}. \end{aligned} \quad (4)$$

where ΔS_t^* and ΔH_t^* are the optimal software and hardware updates; $E_{t:t+k}$ denotes the environment over a prediction horizon of length k ; J

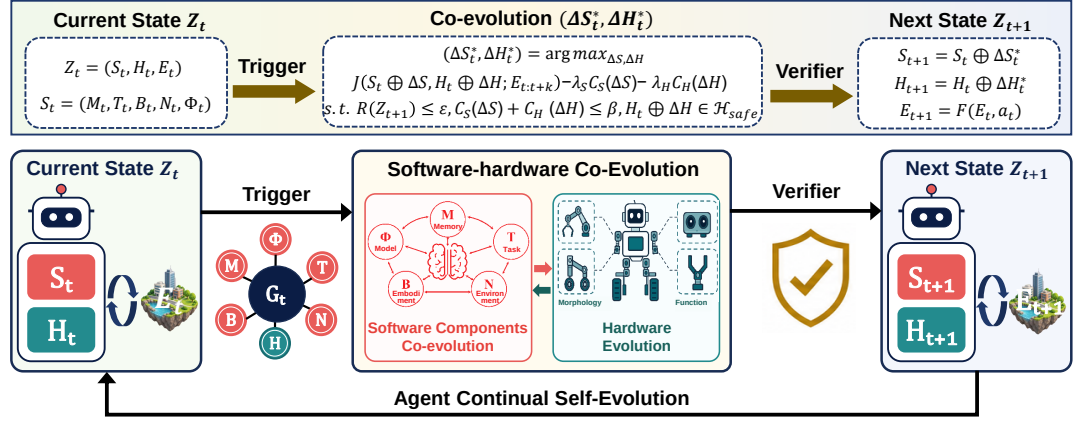


Figure 2. A roadmap for self-evolving embodied AI. At time t , the joint state Z_t comprises the software state S_t , hardware state H_t , and external environment state E_t . Trigger gates G_t activate the required software components and hardware component. The selected software and hardware updates are jointly optimized under performance, cost, risk, and safety constraints and are verified before being deployed to form the next state Z_{t+1} . Repeating this process enables agent continual self-evolution.

measures the expected performance; C_S and C_H measure software and hardware update costs; and λ_S and λ_H are their corresponding weights. The operator $\oplus$ represents an implementation-dependent update. Moreover, R measures risk, ϵ is the maximum acceptable risk, β is the total update budget, and $\mathcal{H}_{safe}$ is the set of safe hardware configurations.

The five software components evolve collaboratively, while software evolution guides physical embodiment evolution and the evolved physical embodiment changes subsequent perception and action. A verifier checks the constraints before the updates are deployed. Once verified, the state transition is:

$$\begin{aligned} S_{t+1} &= S_t \oplus \Delta S_t^*, \\ H_{t+1} &= H_t \oplus \Delta H_t^*, \\ E_{t+1} &= F(E_t, a_t). \end{aligned} \quad (5)$$

where S_{t+1} , H_{t+1} , and E_{t+1} are the next software, hardware, and environment states; a_t is the agent action; and F is the environment transition function. The resulting Z_{t+1} enters the next cycle, enabling agent continual self-evolution.

2.5 Evaluation Indicators

Evaluation indicators for self-evolving embodied AI should measure whether an agent can continuously evolve its software-level and hardware-level components, and evolve the overall agent system through long-horizon interaction. For memory self-updating, evaluation can focus on the effectiveness of memory self-editing, self-organization, and self-distillation, including whether the agent can update, structure, and abstract experiences for later decision-making [36,102]. For task self-switching, in-

dicators should assess task self-selection and task self-generation, measuring whether the agent can select feasible tasks or generate new tasks that promote continuous improvement [39, 64]. For environment self-prediction, evaluation should consider understanding world models, generative world models, and collaborative world models, including the accuracy, consistency, and decision usefulness of latent prediction, future imagination, and world-model collaboration [12,13,75]. For embodiment self-adaptation, indicators should correspond to self-transfer, self-calibration and self-recovery, measuring calibration accuracy and recovery performance under morphology, sensing, actuation, or physical-constraint changes [43,79,83]. For model self-evolution, evaluation should cover model self-restructuring, self-optimization, and self-evaluation, measuring whether architecture, optimization, and evaluation mechanisms can be autonomously improved without unstable degradation [49,85,92]. For physical embodiment evolution, evaluation should cover morphological reconfiguration and functional reconfiguration, measuring whether morphology and function can be safely reconfigured in response to software guidance, as well as whether the underlying architecture of the agent's physical embodiment is altered or its design space is expanded.

At the system level, self-evolving embodied AI can be evaluated by software-hardware co-evolution consistency, autonomous evolution cost, physical embodiment evolution cost, and long-horizon robustness [19,20,27]. Co-evolution gain can further quantify whether coupling software evolution with physical embodiment evolution outperforms software-only evolution or physical embodiment evolution alone.

Table 2. Taxonomy of representative works for self-evolving embodied AI.

Module	Taxonomy	Representative Works
Memory self-updating	Self-editing	SAGE [37], Mem0 [53], DRC [54], Memento [55]
	Self-organization	A-MEM [36], MemInsight [56], MemGen [29], ReMe [57], Generative Agents [58]
	Self-distillation	ExpeL [38], AWM [59], MUSE [60]
Task Self-switching	Self-selection	SEC [39], WebRL [61], Agent0 [62], Mobile-Agent-E [32]
	Self-generation	ZeroGUI [40], AgentEvolver [63], WebEvolver [64]
Environment Self-prediction	Understanding WM	DreamerV3 [12], DreamerV4 [65], JEPA [66], V-JEPA [67], EvolvingAgent [68], ModularAgent [69]
	Generative WM	Genie [13], Genie-2 [70], Matrix-Game-2 [71], MineWorld [72], MineDreamer [73], OA [74]
	Collaborative WM	WorMI [75], COMBO [76], WebEvolver [64], WKM [77]
Embodiment Self-adaptation	Self-transfer	GET-Zero [43], BoT [78], PEAC [79]
	Self-calibration	UP-OSI [45], SPI-Active [80], OFCI [81]
	Self-recovery	Fall Recovery [82], Damage Recovery [83]
Model Self-evolution	Self-restructuring	MaskTAS [47], PMoE [84], MoE-Adapters4CL [85], D-MoLE [86], L2R [87], EEP [88]
	Self-optimization	MoE-CL [48], SRT [89], SEAS [90], TTRL [91], Self-Refine [92], Reflexion [93]
	Self-evaluation	LLM-as-a-Judge [49], SER [94], RLME [95], RLIF [96], MAE [97]
Physical Embodiment Evolution	Morphology	EmbeddedSM [52], PhysicalPM [51], HEXEL [98], SCTB [50]
	Function	M4 [99], UniCore [100], FlexiFly [101]

3. Methodologies

Self-evolving embodied AI comprises five software-level components and a hardware-level component. As shown in Table 2, existing methods realize only partial components, and no system has yet achieved closed-loop co-evolution across both software and hardware levels.

3.1 Memory Self-updating

Existing works on memory self-updating can be broadly categorized into three classes: memory self-editing, memory self-organization, and memory self-distillation. These categories primarily differ in how experience is modified, structured, and abstracted to support long-horizon adaptation over time.

Memory self-editing. Those works focus on explicit operations that directly modify stored memory content, including addition, update, deletion, merging, and forgetting. SAGE [37] introduces reflective mechanisms that regulate memory retention using forgetting curves. Mem0 [53] formalizes memory editing by ex-

tracting salient facts from interaction histories and consolidating them into long-term memory. DRC [54] establishes a closed-loop dual regression framework to tackle ill-posed super-resolution within a constrained search space, effectively refining degraded inputs and providing higher-quality experience for downstream tasks. Memento [55] optimizes memory rewriting and retrieval policies to maintain compact and relevant long-term memory. Together, these works treat memory evolution as an operation-driven editing process.

Memory self-organization. Those works emphasize evolving the structure of memory rather than the content. A-MEM [36] organizes agent memory as a dynamically indexed and linked note network inspired by Zettelkasten-style knowledge management. MemInsight [56] augments raw episodic memory with semantic structure, enabling more robust retrieval and reuse. MemGen [29] explores generative latent memory, where memory is woven into a continuous latent space that evolves with experience. Related systems such as ReMe [57] and Generative

Agents [58] further highlight the role of structured memory in supporting long-term reasoning and consistency. These works view memory evolution as an organizational process.

Memory self-distillation. Those works aim to transform episodic experience into reusable knowledge, skills, or workflows. ExpeL [38] extracts abstract insights and rules from interaction trajectories to guide future behavior. AWM [59] stores reusable task workflows, enabling agents to transfer procedural knowledge across tasks. MUSE [60] further organizes memory hierarchically to accumulate strategic and procedural competence over long horizons. Collectively, these works treat memory evolution as a distillation process that converts episodic experiences into a persistent capability.

3.2 Task Self-switching

Existing works on task self-switching can be broadly categorized into two classes: task self-selection and task self-generation. These two lines differ in whether tasks are adaptively selected from an evolving candidate set or generated online conditioned on the agent's self-state and environment dynamics.

Task self-selection. Those works formulate task switching as a selection or scheduling problem, where an agent dynamically chooses which task to pursue next based on learning progress or interaction feedback. Representative works include Self-Evolving Curriculum (SEC) [39], which learns adaptive task scheduling policies; WebRL [61], which employs a self-evolving online curriculum for web interaction; Agent0 [62], which co-evolves a curriculum agent with an executor; as well as hierarchical embodied systems such as Mobile-Agent-E [32], where high-level managers perform goal or subgoal selection during execution.

Task self-generation. Those works focus on creating new tasks online rather than selecting from a predefined pool, enabling task switching driven by the agent's self-state and exploration demands. ZeroGUI [40] automatically generates GUI tasks from interface states to support online learning; AgentEvolver [63] generates new tasks via self-questioning to guide exploration in novel environments; and WebEvolver [64] leverages a coevolving world model to generate self-instructed interaction tasks.

3.3 Environment Self-prediction

Existing works on environment self-prediction can be broadly categorized into three classes: understanding World Models (WMs), generative world models, and collaborative world models. Understanding world models learn predictive latent representations that support predicting and planning without necessarily decoding raw observations. Generative world models explicitly model the observation-generation process (e.g., next-frame or rollouts) with probabilistic generative models, while collaborative world models integrate multiple predictive modules to support action-conditioned, uncertainty-aware, and long-horizon environment prediction.

Understanding world models. Those works focus on learning predictive latent representations of environment dynamics, typically instantiated via RSSM-style latent dynamics models [12,65] or JEPA-style predictive representation learning [66,67]. These works can achieve environment self-prediction in self-evolving embodied AI by continually evolving latent representations under distribution shifts, e.g., continual world-model updates for long-horizon tasks in EvolvingAgent [68]; task-aware world modeling for navigation in ModularAgent [69].

Generative world models. Those works explicitly predict future environment observations, commonly implemented via autoregressive Transformers or diffusion models. Recent large-scale generative world models, such as Genie [13], Genie-2 [70], and Matrix-Game-2 [71], learn general-purpose environment dynamics from large-scale video data and enable long-horizon imagination. In Minecraft and related open-world environments, interactive generative world models such as MineWorld [72] and MineDreamer [73] provide action-conditioned video generation, and OA [74] plans with online world models in continual RL, forming a practical foundation for environment self-prediction when combined with continual data collection and online refinement.

Collaborative world models. Those works emphasize how multiple world models are composed, retrieved, aligned, or co-evolved to support environment self-prediction. Recent works have explored several forms of world-model collaboration. WorMI [75] incorporates independently learned domain-specific world models into an embodied agent through test-time composition, enabling cross-domain adaptation by retriev-

ing and aligning relevant world-model knowledge. COMBO [76] learns compositional world models for embodied multi-agent cooperation, where partial egocentric observations and multiple agents' actions are integrated to simulate joint future dynamics for online planning. WebEvolver [64] introduces a co-evolving World Model that collaborates with the agent policy by generating self-instructed training data and providing look-ahead imagination during inference. WKM [77] further shows that multiple world models can provide both global prior and local dynamic knowledge to facilitate agent planning.

3.4 Embodiment Self-adaptation

Existing works on embodiment self-adaptation can be broadly categorized into three classes: embodiment self-transfer, embodiment self-calibration, and embodiment self-recovery. This software-level component adapts internal representations to changes in the physical embodiment without modifying the hardware itself.

Embodiment self-transfer. Those works transfer software policies or representations across different given physical embodiments. Representative works include GET-Zero [43], which models embodiment topology for zero-shot generalization; Body Transformer (BoT) [78], which incorporates embodiment structure into policy learning; and PEAC [79], which pretrains transferable representations for cross-embodiment control.

Embodiment self-calibration. Those works target continuous changes in actuation limits and physical constraints, such as mass variation, friction, contact dynamics, and sensing noise. This line typically relies on online system identification or latent parameter estimation to recalibrate internal embodiment models during execution. A classical foundation is the universal policy with online system identification (UP-OSI) [45], with recent extensions enabling active or continual calibration in contact-rich scenarios, such as SPI-Active [80] and Online Friction Coefficient Identification (OFCI) [81] for legged robots.

Embodiment self-recovery. Those works address sudden degradation or failure of sensing, actuation, or computational resources, requiring rapid restoration of executable behavior. This line emphasizes fault-aware adaptation and recovery rather than parameter refinement. Representative works include quality-diversity based damage recovery [83], which maintains diverse

behavioral repertoires for fast adaptation under damage, and learning-based fall or failure recovery policies for legged robots [82].

3.5 Model Self-evolution

Existing works on model self-evolution can be broadly categorized into three classes: model self-restructuring, model self-optimization and model self-evaluation.

Model self-restructuring. Those works evolve the internal architecture of models through modular growth, expert composition, or adaptive routing to meet changing requirements across embodiments, environments, and tasks. Representative works include self-supervised architecture search via masked distillation [47], adaptive Mixture of Expert (MoE) models for continual learning [84,85], dynamic adapter or LoRA expert composition [86,87], and modular growth with expert pruning [88].

Model self-optimization. Those works focus on autonomously improving training and parameter update strategies through continual tuning, self-generated feedback, or test-time learning. Typical works include continual instruction tuning without human labels [48], learning from language feedback in SRT [89], adversarial self-improvement in SEAS [90], test-time reinforcement learning [91], and iterative self-refinement and reflection-based optimization [92,93]. Within test-time learning, EATA-C [103] shows that reliable test-time self-adaptation depends on selecting trustworthy unlabeled samples, pioneering entropy-based criteria to filter ambiguous or harmful inputs for self-optimization.

Model self-evaluation. Those works evolve evaluation criteria and reward signals to guide learning in the absence of reliable ground-truth supervision. This line is characterized by LLM-as-a-judge frameworks [49], self-evolved reward learning [94], reinforcement learning from meta-evaluation [95], learning without external rewards [96], and multi-judge co-evolution [97].

3.6 Physical Embodiment Evolution

Existing works on physical embodiment evolution mainly focus on morphological reconfiguration and functional reconfiguration. The former changes the agent's physical form, while the latter changes its physical sensing or actuation ca-

pabilities.

Morphological reconfiguration. Those works change the embodiment shape or mechanical organization of an agent. EmbeddedSM [52] changes embodiment and limb geometry through integrated actuation and shape locking; physical polygon meshing (PhysicalPM) [51] changes module geometry and connectivity; shape-changing tensegrity blocks (SCTB) [50] combine deformation and modular assembly to construct different physical structures; and HEXEL [98] rearranges electrohydraulic modules through reversible magnetic connections to construct robotic systems with different physical morphologies and locomotion configurations.

Functional reconfiguration. Those works change physical functions by replacing, rearranging, or repurposing sensing and actuation components. M4 [99] repurposes appendages as wheels, legs, hands, or thrusters; UniCore [100] combines a shared actuation core with interchangeable limbs; and FlexiFly [101] attaches task-relevant sensing and actuation modules under software guidance.

4. Applications

This section discusses three practical applications of self-evolving embodied AI: autonomous service robotics, autonomous driving, and autonomous Unmanned Aerial Vehicles (UAVs). A case study further illustrates the co-evolution between software and hardware.

4.1 Autonomous Service Robotics

Autonomous service robots operate in open-ended human-centered environments where object layouts, user preferences, and task specifications change over time [104]. Recent advances in open-vocabulary mobile manipulation and generalist robot policies demonstrate strong generalization across tasks and scenes [105–107], but comprehensive deployment still requires self-evolution beyond static task execution.

In this scenario, the priority software-level components are memory self-updating, task self-switching, and embodiment self-adaptation. Memory self-updating retains user preferences and failure cases, task self-switching adjusts goals under changing instructions, and embodiment self-adaptation adjusts software policies across heterogeneous platforms. Physical embodiment evolution may involve reconfiguring

manipulators, sensors, or mobility modules, with the resulting hardware changes providing feedback for subsequent software evolution. Existing studies still rely heavily on fixed tasks, short-term memory, and platform-specific policies, leaving long-horizon adaptation and software-hardware co-evolution insufficiently explored.

4.2 Autonomous Driving

Autonomous driving represents a safety-critical instantiation of embodied AI [108] in dynamic environments. Real-world traffic exhibits non-stationary dynamics, rare corner cases, and complex interactions with humans [109], which cannot be exhaustively captured during training. EPMF [110] attributes perception failures to modality discrepancies and proposes discrepancy-aware modality selection and fusion, improving the reliability of multi-sensor 3D semantic segmentation. Recent work increasingly relies on predictive world models [111] and high-level reasoning [112] to support planning and simulation.

In this scenario, the priority software-level components are environment self-prediction, model self-evolution, and embodiment self-adaptation. Environment self-prediction anticipates future trajectories and risks, model self-evolution maintains robustness under distribution shifts, and embodiment self-adaptation handles sensor calibration, vehicle dynamics, and hardware degradation. In safety-critical settings, physical embodiment evolution should be restricted to validated changes in sensing or actuation configurations, with corresponding software recalibration. Existing research is still largely evaluated on static datasets or closed benchmarks, often emphasizing short-horizon prediction rather than long-horizon evolution.

4.3 Autonomous UAVs

Autonomous UAVs [113,114] operate under fast dynamics, partial observability, and strict physical constraints, making them a natural testbed for self-evolving embodied AI. UAV missions such as inspection [115], planning [116], and search-and-rescue [117] require continual adaptation to dynamic environments [118], weather conditions [119], and task objectives [120].

In this scenario, the priority software-level components are embodiment self-adaptation, environment self-prediction, and memory self-updating. For modular UAVs [101], physical embodiment evolution may involve reconfiguring payload, sensing, or actuation modules,

while the software adapts its perception and control policies to the resulting hardware configuration. Environment self-prediction supports wind-field estimation and obstacle-motion prediction, while memory self-updating retains rare but safety-critical experiences. These processes are constrained by onboard computation, battery capacity, communication bandwidth, and real-time control requirements. Existing studies often assume fixed physical embodiments or simulation-only evaluation, limiting the verification of long-horizon software-hardware co-evolution.

4.4 Case Study

Consider a home service robot asked to “prepare a cup of tea for the user” in a changing household environment. The agent first performs memory self-updating by retrieving and updating user preferences, object locations, and past failure cases [36,58]. Then, task self-switching decomposes the goal into feasible sub-tasks, such as finding the kettle, checking water, grasping the cup, and delivering it to the user, based on language-grounded affordances and open-ended task exploration [39,62,63]. Meanwhile, environment self-prediction estimates future states and action consequences, such as whether opening a cabinet may reveal the target cup or whether a path will be blocked [12,64]. During execution, embodiment self-adaptation adjusts software policies according to the robot’s physical constraints [43,78,79]. If software adaptation is insufficient, physical embodiment evolution can replace an end effector or sensing module. Finally, model self-evolution refines policies, reasoning procedures, or action representations from feedback and failures [84,85,88]. The five software-level components co-evolve with each other, while hardware reconfiguration and subsequent software adaptation form a software-hardware co-evolution loop.

Current systems mainly evaluate predefined task completion rather than long-horizon co-evolution among software-level components and reconfigurable hardware [105]. Most methods still depend on fixed physical embodiments and improve isolated capabilities, making controllable software-level and software-hardware co-evolution difficult to verify.

5. Future Directions

Self-evolving embodied AI presents a new paradigm for moving embodied AI beyond human-crafted setting to in-the-wild setting. Fu-

ture research should focus on the controllability, trustworthiness, and swarm-level systems.

5.1 Controllable Self-evolution

A core challenge for self-evolving embodied AI is ensuring that the self-evolution process remains controllable under real-world constraints. Uncontrolled software evolution or physical embodiment evolution may cause unstable updates, unsafe actions, or unintended changes in the agent’s objectives or physical configuration.

At the action level, evolved policies should satisfy collision, actuation, energy, and task-specific safety constraints. At the component level, software evolution should be regulated by trigger thresholds and adaptation budgets, while physical embodiment evolution should require feasibility and safety checks. At the system level, fast perception-action control should be separated from slower software evolution and physical embodiment evolution. Quantifiable criteria such as stability margins and risk bounds will be essential for predictable operation.

5.2 Trustworthy Self-evolution

The applications of self-evolving embodied AI to real-world systems require self-evolution to be safe, transparent, accountable, and aligned with human values. Continuous monitoring and verification should ensure that evolved software and evolved physical embodiments do not produce unsafe or misaligned behavior.

Software updates and physical embodiment evolution should be evaluated through simulation, hardware-in-the-loop testing, uncertainty estimation, and safety monitoring. Evolution logs should record both software and hardware changes, while rollback mechanisms should restore safe software and hardware configurations when failures occur.

Human-AI alignment should be embedded into self-evolution rather than treated as an external constraint. Future work should incorporate human oversight, preference constraints, and approval mechanisms for high-risk changes. In safety-critical applications, changes to task objectives, model structures, or physical embodiments should require stronger verification and human authorization.

5.3 Swarm Self-evolution

While existing studies primarily focus on individual agents, many applications involve multiple agents operating collectively. Swarm self-

evolution requires agents to share experiences, coordinate software evolution, and allocate or reconfigure hardware functions under resource constraints. Future research should explore distributed memory, policy sharing, role differentiation, and hardware-resource allocation. Understanding emergent behaviors and aligning individual evolution with swarm-level objectives are also crucial. Collective verification is necessary to prevent local software or hardware changes from causing globally unsafe behavior.

6. Conclusion

This paper presents self-evolving embodied AI as a new paradigm for moving embodied AI from human-crafted setting toward in-the-wild setting. It comprises five software-level components and a hardware-level component for physical embodiment evolution. Co-evolution among software-level components and software-hardware co-evolution jointly support continually adaptive intelligence with autonomous evolution. This framework incorporates hardware into self-evolving embodied AI while distinguishing physical embodiment evolution from software-level embodiment self-adaptation. We believe that self-evolving embodied AI can provide a new perspective toward general artificial intelligence.

FUNDING

This work was supported by Beijing National Research Center for Information Science and Technology under Grant No.BNR2023TD03006.

AUTHOR CONTRIBUTIONS

T.F., X.W., and W.Z. conceived the work. T.F., X.W., and W.Z. contributed to the definition, framework, components, mechanisms, and roadmap of self-evolving embodied AI. T.F. and X.W. contributed to the review of state-of-the-art works for realized components. T.F., X.W., and W.Z. contributed to the writing of this paper. W.Z. supervised the whole project.

Conflict of interest statement. None declared.

REFERENCES

1. Turing AM. Computing machinery and intelligence. *Parsing the Turing test: Philosophical and methodological issues in the quest for the thinking computer*, 23–65 (2007).

2. Feng T, Wang X, Jiang YG *et al*. Embodied ai: From llms to world models [feature]. *IEEE Circuits and Systems Magazine* 2025; **25**: 14–37.
3. Sun F, Chen R, Ji T *et al*. A comprehensive survey on embodied intelligence: Advancements, challenges, and future perspectives. *CAAI Artificial Intelligence Research* 2024; **3**: 1.
4. Xu Z, Wu K, Wen J *et al*. A survey on robotics with foundation models: toward embodied ai. *arXiv preprint arXiv:2402.02385* 2024; .
5. Ma Y, Song Z, Zhuang Y *et al*. A survey on vision-language-action models for embodied ai. *arXiv preprint arXiv:2405.14093* 2024; .
6. Hu Y, Xie Q, Jain V *et al*. Toward general-purpose robots via foundation models: A survey and meta-analysis. *arXiv preprint arXiv:2312.08782* 2023; .
7. Duan J, Yu S, Tan HL *et al*. A survey of embodied ai: From simulators to research tasks. *IEEE Transactions on Emerging Topics in Computational Intelligence* 2022; **6**: 230–244.
8. Srivastava S, Li C, Lingelbach M *et al*. Behavior: Benchmark for everyday household activities in virtual, interactive, and ecological environments. *Conference on robot learning* (2022) 477–490.
9. Brohan A, Brown N, Carbajal J *et al*. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817* 2022; .
10. O'Neill A, Rehman A, Maddukuri A *et al*. Open x-embodiment: Robotic learning datasets and rt-x models. *2024 IEEE International Conference on Robotics and Automation (ICRA)* (2024) 6892–6903.
11. Zitkovich B, Yu T, Xu S *et al*. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *Conference on Robot Learning* (2023) 2165–2183.
12. Hafner D, Pasukonis J, Ba J *et al*. Mastering diverse control tasks through world models. *Nature* 2025; 1–7.
13. Bruce J, Dennis MD, Edwards A *et al*. Genie: Generative interactive environments. *Forty-first International Conference on Machine Learning* (2024) .
14. Liu Y, Chen W, Bai Y *et al*. Aligning cyber space with physical world: A comprehensive survey on embodied ai. *IEEE/ASME Transactions on Mechatronics* 2025; .
15. Zhu D, Bu Q, Zhu Z *et al*. Advancing autonomy through lifelong learning: a survey of autonomous intelligent systems. *Frontiers in neurorobotics* 2024; **18**: 1385778.
16. Wang L, Zhang X, Su H *et al*. A comprehensive survey of continual learning: Theory, method and application. *IEEE transactions on pattern analysis and machine intelligence* 2024; **46**: 5362–5383.
17. Charlesworth WR. Human intelligence as adaptation: An ethological approach. *The nature of intelligence* (2024) 147–168.
18. Wang X, Feng T, Liu H *et al*. *Embodied AI*, 3–88 (2026).
19. Fang J, Peng Y, Zhang X *et al*. A comprehensive survey of self-evolving ai agents: A new paradigm bridging foundation models and lifelong agentic systems. *arXiv preprint arXiv:2508.07407* 2025; .
20. Gao Ha, Geng J, Hua W *et al*. A survey of self-evolving agents: On path to artificial super intelligence. *arXiv preprint arXiv:2507.21046* 2025; .
21. Wang X, Feng T, Liu H *et al*. *Self-evolving Agent*, 247–267 (2026).

22. Wang X, Feng T, Liu H *et al.* *Self-evolving Embodied Intelligence*, 189–209 (2026).
23. Wang X, Feng T, Liu H *et al.* *Autonomous Embodied AI*, 89–121 (2026).
24. Wang X, Feng T, Liu H *et al.* *Autonomous Swarm Embodied AI*, 123–186 (2026).
25. Ou Y, Zhou W, Ding S *et al.* Symbolic learning enables self-evolving agents. *AI Open* 2025; .
26. Yadav S and Roy L. Adaptive prompt intelligence: Towards self-evolving conversational agents. *2025 IEEE 5th International Conference on ICT in Business Industry & Government (ICTBIG)* (2025) 1–6.
27. Cai Y, Hao Y, Zhou J *et al.* Building self-evolving agents via experience-driven lifelong learning: A framework and benchmark. *arXiv preprint arXiv:2508.19005* 2025; .
28. Xu H, Hu J, Zhang K *et al.* Sedm: Scalable self-evolving distributed memory for agents. *arXiv preprint arXiv:2509.09498* 2025; .
29. Zhang G, Fu M and Yan S. Memgen: Weaving generative latent memory for self-evolving agents. *arXiv preprint arXiv:2509.24704* 2025; .
30. Huang C, Yu W, Wang X *et al.* R-zero: Self-evolving reasoning llm from zero data. *arXiv preprint arXiv:2508.05004* 2025; .
31. Liu J, Xiong K, Xia P *et al.* Agent0-vl: Exploring self-evolving agent for tool-integrated vision-language reasoning. *arXiv preprint arXiv:2511.19900* 2025; .
32. Wang Z, Xu H, Wang J *et al.* Mobile-agent-e: Self-evolving mobile assistant for complex tasks. *arXiv preprint arXiv:2501.11733* 2025; .
33. Thawakar O, Venkatraman S, Thawkar R *et al.* Evolmm: Self-evolving large multimodal models with continuous rewards. *arXiv preprint arXiv:2511.16672* 2025; .
34. Zhang DC, Zhao Y, Wu J *et al.* Evolvesearch: An iterative self-evolving search agent. *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (2025) 13134–13147.
35. Sun Z, Liu Z, Zang Y *et al.* Seagent: Self-evolving computer use agent with autonomous learning from experience. *arXiv preprint arXiv:2508.04700* 2025; .
36. Xu W, Liang Z, Mei K *et al.* A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110* 2025; .
37. Liang X, He Y, Xia Y *et al.* Self-evolving agents with reflective and memory-augmented abilities. *arXiv preprint arXiv:2409.00872* 2024; .
38. Zhao A, Huang D, Xu Q *et al.* Expel: Llm agents are experiential learners. *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38 (2024) 19632–19642.
39. Chen X, Lu J, Kim M *et al.* Self-evolving curriculum for llm reasoning. *arXiv preprint arXiv:2505.14970* 2025; .
40. Yang C, Su S, Liu S *et al.* Zerogui: Automating online gui learning at zero human cost. *arXiv preprint arXiv:2505.23762* 2025; .
41. Wang X, Feng T, Liu H *et al.* *Self-evolving World Model*, 211–245 (2026).
42. Liu H, Huang X, Gu J *et al.* Tcdformer-based momentum transfer model for long-term sports prediction. *Expert Systems with Applications* 2025; **289**: 128310.
43. Patel A and Song S. Get-zero: Graph embodiment transformer for zero-shot embodiment generalization. *2025 IEEE International Conference on Robotics and Automation (ICRA)* (2025) 14262–14269.
44. Shen Y, Liu H, Pei K *et al.* Metaworld: Skill transfer and composition in a hierarchical world model for grounding high-level instructions. *arXiv preprint arXiv:2601.17507* 2026; .
45. Yu W, Tan J, Liu CK *et al.* Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453* 2017; .
46. Wang R, Wang X, Feng T *et al.* Improving compositional generalization in cross-embodiment learning via mixture of disentangled prototypes. *Proceedings of the 33rd ACM International Conference on Multimedia* (2025) 7162–7171.
47. Yan C, Chang X, Li Z *et al.* Masked distillation advances self-supervised transformer architecture search. *The Twelfth International Conference on Learning Representations* (2024) .
48. Kang J, Huang L, Hou C *et al.* Self-evolving llms via continual instruction tuning. *arXiv preprint arXiv:2509.18133* 2025; .
49. Zheng L, Chiang WL, Sheng Y *et al.* Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems* 2023; **36**: 46595–46623.
50. Zhao L, Jiang Y, Chen M *et al.* Modular shape-changing tensegrity-blocks enable self-assembling robotic structures. *Nature communications* 2025; **16**: 5888.
51. Belke CH, Holdcroft KA, Sigrist A *et al.* Morphological flexibility in robotic systems through physical polygon meshing 2023; .
52. Sun J, Lerner E, Tighe B *et al.* Embedded shape morphing for morphologically adaptive robots. *Nature communications* 2023; **14**: 6023.
53. Chhikara P, Khant D, Aryan S *et al.* Mem0: building production-ready ai agents with scalable long-term memory (2025). URL <https://arxiv.org/abs/2504.19413> 2025; .
54. Guo Y, Tan M, Deng Z *et al.* Towards lightweight super-resolution with dual regression learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2024; **46**: 8365–8379.
55. Zhou H, Chen Y, Guo S *et al.* Memento: Fine-tuning llm agents without fine-tuning llms, 2025. URL <https://arxiv.org/abs/2508.16153> ; .
56. Salama R, Cai J, Yuan M *et al.* Meminsight: Autonomous memory augmentation for llm agents. *arXiv preprint arXiv:2503.21760* 2025; .
57. Cao Z, Deng J, Yu L *et al.* Remember me, refine me: A dynamic procedural memory framework for experience-driven agent evolution. *arXiv preprint arXiv:2512.10696* 2025; .
58. Park JS, O'Brien J, Cai CJ *et al.* Generative agents: Interactive simulacra of human behavior. *Proceedings of the 36th annual acm symposium on user interface software and technology* (2023) 1–22.
59. Wang ZZ, Mao J, Fried D *et al.* Agent workflow memory. *arXiv preprint arXiv:2409.07429* 2024; .
60. Yang C, Yang X, Wen L *et al.* Learning on the job: An experience-driven self-evolving agent for long-horizon tasks. *arXiv preprint arXiv:2510.08002* 2025; .
61. Qi Z, Liu X, long IL *et al.* Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning. *arXiv preprint arXiv:2411.02337* 2024; .

62. Xia P, Zeng K, Liu J *et al.* Agent0: Unleashing self-evolving agents from zero data via tool-integrated reasoning. *arXiv preprint arXiv:2511.16043* 2025; .
63. Zhai Y, Tao S, Chen C *et al.* Agentevolver: Towards efficient self-evolving agent system. *arXiv preprint arXiv:2511.10395* 2025; .
64. Fang T, Zhang H, Zhang Z *et al.* Webevolver: Enhancing web agent self-improvement with coevolving world model. *arXiv preprint arXiv:2504.21024* 2025; .
65. Hafner D, Yan W and Lillicrap T. Training agents inside of scalable world models. *arXiv preprint arXiv:2509.24527* 2025; .
66. Assran M, Duval Q, Misra I *et al.* Self-supervised learning from images with a joint-embedding predictive architecture. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023) 15619–15629.
67. Assran M, Bardes A, Fan D *et al.* V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985* 2025; .
68. Feng T, Wang X, Zhou Z *et al.* Evolvingagent: Curriculum self-evolving agent with continual world model for long-horizon tasks. *arXiv preprint arXiv:2502.05907* 2025; .
69. Zhan YW, Wang X, Mao P *et al.* Modularagent: A task-aware modular framework for joint optimization of multimodal large language models and world models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2026) 8087–8096.
70. Parker-Holder J, Ball P, Bruce J *et al.* Genie 2: A large-scale foundation world model. URL: <https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model> 2024; .
71. He X, Peng C, Liu Z *et al.* Matrix-game 2.0: An open-source real-time and streaming interactive world model. *arXiv preprint arXiv:2508.13009* 2025; .
72. Guo J, Ye Y, He T *et al.* Mineworld: a real-time and open-source interactive world model on minecraft. *arXiv preprint arXiv:2504.08388* 2025; .
73. Zhou E, Qin Y, Yin Z *et al.* Minedreamer: Learning to follow instructions via chain-of-imagination for simulated-world control. *arXiv preprint arXiv:2403.12037* 2024; .
74. Liu Z, Fu G, Du C *et al.* Continual reinforcement learning by planning with online world models. *arXiv preprint arXiv:2507.09177* 2025; .
75. Yoo M, Jang J, Yoon S *et al.* World model implanting for test-time adaptation of embodied agents. *arXiv preprint arXiv:2509.03956* 2025; .
76. Zhang H, Wang Z, Lyu Q *et al.* Combo: Compositional world models for embodied multi-agent cooperation. *International Conference on Learning Representations*, volume 2025 (2025) 49996–50019.
77. Qiao S, Fang R, Zhang N *et al.* Agent planning with world knowledge model. *Advances in Neural Information Processing Systems* 2024; **37**: 114843–114871.
78. Sterrazza C, Huang DM, Liu F *et al.* Body transformer: Leveraging robot embodiment for policy learning. *arXiv preprint arXiv:2408.06316* 2024; .
79. Ying C, Hao Z, Zhou X *et al.* Peac: Unsupervised pre-training for cross-embodiment reinforcement learning. *Advances in Neural Information Processing Systems* 2024; **37**: 54632–54669.
80. Sobanbabu N, He G, He T *et al.* Sampling-based system identification with active exploration for legged robot sim2real learning. *arXiv preprint arXiv:2505.14266* 2025; .
81. Kim H, Kang D, Kim MG *et al.* Online friction coefficient identification for legged robots on slippery terrain using smoothed contact gradients. *IEEE Robotics and Automation Letters* 2025; .
82. Yang C, Pu C, Xin G *et al.* Learning complex motor skills for legged robot fall recovery. *IEEE Robotics and Automation Letters* 2023; **8**: 4307–4314.
83. Allard M, Smith SC, Chatzilygeroudis K *et al.* Online damage recovery for physical robots with hierarchical quality-diversity. *ACM Transactions on Evolutionary Learning* 2023; **3**: 1–23.
84. Jung MJ and Kim J. Pmo: Progressive mixture of experts with asymmetric transformer for continual learning. *arXiv preprint arXiv:2407.21571* 2024; .
85. Yu J, Zhuge Y, Zhang L *et al.* Boosting continual learning of vision-language models via mixture-of-experts adapters. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024) 23219–23230.
86. Ge C, Wang X, Zhang Z *et al.* Dynamic mixture of curriculum lora experts for continual multimodal instruction tuning. *arXiv preprint arXiv:2506.11672* 2025; .
87. Araujo V, Moens MF and Tuytelaars T. Learning to route for dynamic adapter composition in continual learning with language models. *Findings of the Association for Computational Linguistics: EMNLP 2024* (2024) 687–696.
88. Liu E, Zhu J, Lin Z *et al.* Efficient expert pruning for sparse mixture-of-experts language models: Enhancing performance and reducing inference costs. *arXiv preprint arXiv:2407.00945* 2024; .
89. Hu C, Hu Y, Cao H *et al.* Teaching language models to self-improve by learning from language feedback. *arXiv preprint arXiv:2406.07168* 2024; .
90. Diao M, Li R, Liu S *et al.* Seas: Self-evolving adversarial safety optimization for large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39 (2025) 23778–23786.
91. Zuo Y, Zhang K, Sheng L *et al.* Ttrl: Test-time reinforcement learning. *arXiv preprint arXiv:2504.16084* 2025; .
92. Madaan A, Tandon N, Gupta P *et al.* Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems* 2023; **36**: 46534–46594.
93. Shinn N, Cassano F, Gopinath A *et al.* Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems* 2023; **36**: 8634–8652.
94. Huang C, Fan Z, Wang L *et al.* Self-evolved reward learning for llms. *arXiv preprint arXiv:2411.00418* 2024; .
95. Rentschler M and Roberts J. Reinforcement learning from meta-evaluation: Aligning language models without ground-truth labels. *arXiv preprint arXiv:2601.21268* 2026; .
96. Zhao X, Kang Z, Feng A *et al.* Learning to reason without external rewards. *arXiv preprint arXiv:2505.19590* 2025; .
97. Chen Y, Wang Y, Zhu S *et al.* Multi-agent evolve: Llm self-improve through co-evolution. *arXiv preprint arXiv:2510.23595* 2025; .

98. Yoder Z, Rumley EH, Schmidt I *et al.* Hexagonal electro-hydraulic modules for rapidly reconfigurable high-speed robots. *Science Robotics* 2024; **9**: eadl3546.
99. Sihite E, Kalantari A, Nemovi R *et al.* Multi-modal mobility morphobot (m4) with appendage repurposing for locomotion plasticity enhancement. *Nature communications* 2023; **14**: 3323.
100. Li S, Liu F, Dong X *et al.* Reciprocal actuation core and modular robotic limbs for flying, swimming and running. *Communications Engineering* 2025; **4**: 71.
101. Zhao M, Xia J, Hou K *et al.* Flexify: interfacing the physical world with foundation models empowered by reconfigurable drone systems. *Proceedings of the 23rd ACM Conference on Embedded Networked Sensor Systems* (2025) 463–476.
102. Yan S, Yang X, Huang Z *et al.* Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. *arXiv preprint arXiv:2508.19828* 2025; .
103. Tan M, Chen G, Wu J *et al.* Uncertainty-calibrated test-time model adaptation without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2025; .
104. Zhang W, Li Y, Qiao Y *et al.* Effective tuning strategies for generalist robot manipulation policies. *2025 IEEE International Conference on Robotics and Automation (ICRA)* (2025) 7255–7262.
105. Yenamandra S, Ramachandran A, Yadav K *et al.* Homerobot: Open-vocabulary mobile manipulation. *arXiv preprint arXiv:2306.11565* 2023; .
106. Huang S, Chen L, Zhou P *et al.* Enerverse: Envisioning embodied future space for robotics manipulation. *arXiv preprint arXiv:2501.01895* 2025; .
107. Driess D, Xia F, Sajjadi MS *et al.* Palm-e: An embodied multimodal language model 2023; .
108. Yurtsever E, Lambert J, Carballo A *et al.* A survey of autonomous driving: Common practices and emerging technologies. *IEEE access* 2020; **8**: 58443–58469.
109. Schwarting W, Alonso-Mora J and Rus D. Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems* 2018; **1**: 187–210.
110. Tan M, Zhuang Z, Chen S *et al.* Epmf: Efficient perception-aware multi-sensor fusion for 3d semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2024; **46**: 8258–8273.
111. Peng Q, Bai C, Zhang G *et al.* Navigscene: Bridging local perception and global navigation for beyond-visual-range autonomous driving. *Proceedings of the 33rd ACM International Conference on Multimedia* (2025) 4193–4202.
112. Sun W, Lin X, Shi Y *et al.* Sparsedrive: End-to-end autonomous driving via sparse scene representation. *2025 IEEE International Conference on Robotics and Automation (ICRA)* (2025) 8795–8801.
113. Chang Y, Cheng Y, Manzoor U *et al.* A review of uav autonomous navigation in gps-denied environments. *Robotics and Autonomous Systems* 2023; **170**: 104533.
114. Feng T, Wang X, Han F *et al.* U2udata: A large-scale cooperative perception dataset for swarm uavs autonomous flight. *Proceedings of the 32nd ACM International Conference on Multimedia* (2024) 7600–7608.
115. Li Z, Zhang Y, Wu H *et al.* Design and application of a uav autonomous inspection system for high-voltage power transmission lines. *Remote Sensing* 2023; **15**: 865.
116. Deng W, Feng J and Zhao H. Autonomous path planning via sand cat swarm optimization with multi-strategy mechanism for unmanned aerial vehicles in dynamic environment. *IEEE Internet of Things Journal* 2025; .
117. Feng T, Wang X, Han F *et al.* U2udata+: A scalable swarm uavs autonomous flight dataset for embodied long-horizon tasks. *Thirtieth AAAI Conference on Artificial Intelligence* 2026; .
118. Wang C, Wang J, Shen Y *et al.* Autonomous navigation of uavs in large-scale complex environments: A deep reinforcement learning approach. *IEEE Transactions on Vehicular Technology* 2019; **68**: 2124–2136.
119. Feng T, Li Q, Wang X *et al.* Multi-weather cross-view geolocalization using denoising diffusion models. *Proceedings of the 2nd Workshop on UAVs in Multimedia: Capturing the World from a New Perspective* (2024) 35–39.
120. Du Z, Luo C, Min G *et al.* A survey on autonomous and intelligent swarms of uncrewed aerial vehicles (uavs). *IEEE Transactions on Intelligent Transportation Systems* 2025; .