Tsinghua Science and Technology
Doi: 10.26599/TST.2026.9010072

Review Article

A survey of large-model-based AI agents

Chuan Qin and Long Jin (✉)

Abstract Advances of large models (LMs) have catalyzed a paradigm shift in artificial intelligence, enabling the development of autonomous agents capable of complex reasoning, planning, and interaction with both digital and physical environments. As this field has expanded at an unprecedented rate, a comprehensive and structured overview is essential to consolidate current knowledge and guide future innovations. This survey addresses this need by providing a holistic review of LM-based artificial intelligence (AI) agents. First, we deconstruct the core architecture of modern LM-based agents and examine the interplay among key modules: Reasoning, perception, memory, planning, action, and learning. Subsequently, we systematically analyze the evaluation landscape, summarizing current benchmarks, metrics, and module-specific performance trade-offs. Furthermore, we survey the transformative impact of these agents across a broad spectrum of applications, ranging from digital domains to embodied systems. The survey concludes by identifying critical challenges and future directions, thus offering a roadmap for the next generation of LM-based AI agents.

Keywords Artificial intelligence (AI), large models (LMs), AI agents, embodied AI, deep learning

1 Introduction

The advance of large models (LMs) has catalyzed a watershed moment in artificial intelligence (AI). While initially driven by large language models (LLMs), the cognitive core of these systems is rapidly expanding to include multimodal large models (MLLMs) and vision-language-action (VLA) models. This marks a paradigm shift from models that passively generate content to autonomous agents capable of actively reasoning, perceiving, planning, and acting within complex environments [1–3]. LM-based AI agents are systems capable of interpreting high-level human goals, autonomously decomposing them into actionable steps, and executing those steps by intelligently interacting with a vast array of digital or physical tools [4]. This progress from specialized, task-specific models to generalist, goal-oriented agents represents a fundamental rethinking of how intelligent systems are built and used, unlocking new frontiers in automation, scientific discovery, and human-computer collaboration. The development of these agents builds upon decades of foundational research on the dynamics and stability of various neural network architectures [5]. For instance, early analyses on the stability of recurrent and cellular neural networks with time-varying delays established key theoretical principles for building reliable intelligent systems [6, 7].

As this field expands at an unprecedented rate, a torrent of new architectures, methodologies, and applications emerges almost daily. Concepts such as memory systems [8], tool augmentation [9], and self-reflection [10] are rapidly evolving, leading to a vibrant but fragmented research landscape. For both seasoned researchers and newcomers, navigating this deluge of information to discern foundational principles from fleeting trends has become a formidable challenge. A comprehensive and structured overview is therefore not merely beneficial but essential for consolidating current knowledge, establishing a common conceptual framework, and guiding future innovations in a cohesive direction.

Prior work has surveyed parts of this landscape from distinct angles, but each leaves concrete white space that our survey fills. Xi et al. [1] organize agents around a three-part Brain–Perception–Action decomposition, situating them within single-agent, multi-agent, and agent-society scenarios. Reasoning is subsumed inside the Brain module rather than treated as a coordinating core, and the survey lacks a dedicated learning/reflection module. Embodied applications are only briefly mentioned. Durante et al. [4] adopt a Multimodal interaction perspective, covering gaming, robotics, and healthcare through the agent Transformer view. The taxonomy is thematic rather than architectural, with no module-level decomposition, and important digital domains such as graphical user interface (GUI)/web, code, math, and autonomous driving are not addressed. Wang et al. [11] introduce the influential Profile–

• Chuan Qin is with the School of Information Science and Engineering, Lanzhou University, Lanzhou 730000, China, and also with the College of Computer Science and Engineering, Jishou University, Jishou 416000, China. E-mail: qinchuan8@foxmail.com.
• Long Jin is with the School of Information Science and Engineering, Lanzhou University, Lanzhou 730000, China, and also with the School of Automation and Electrical Engineering, Lanzhou University of Technology, Lanzhou 730050, China. E-mail: jinlong@lzu.edu.cn.
* To whom correspondence should be addressed.
Manuscript received: 20xx-xx-xx; revised: 20xx-11-xx; accepted: 20xx-00-xx

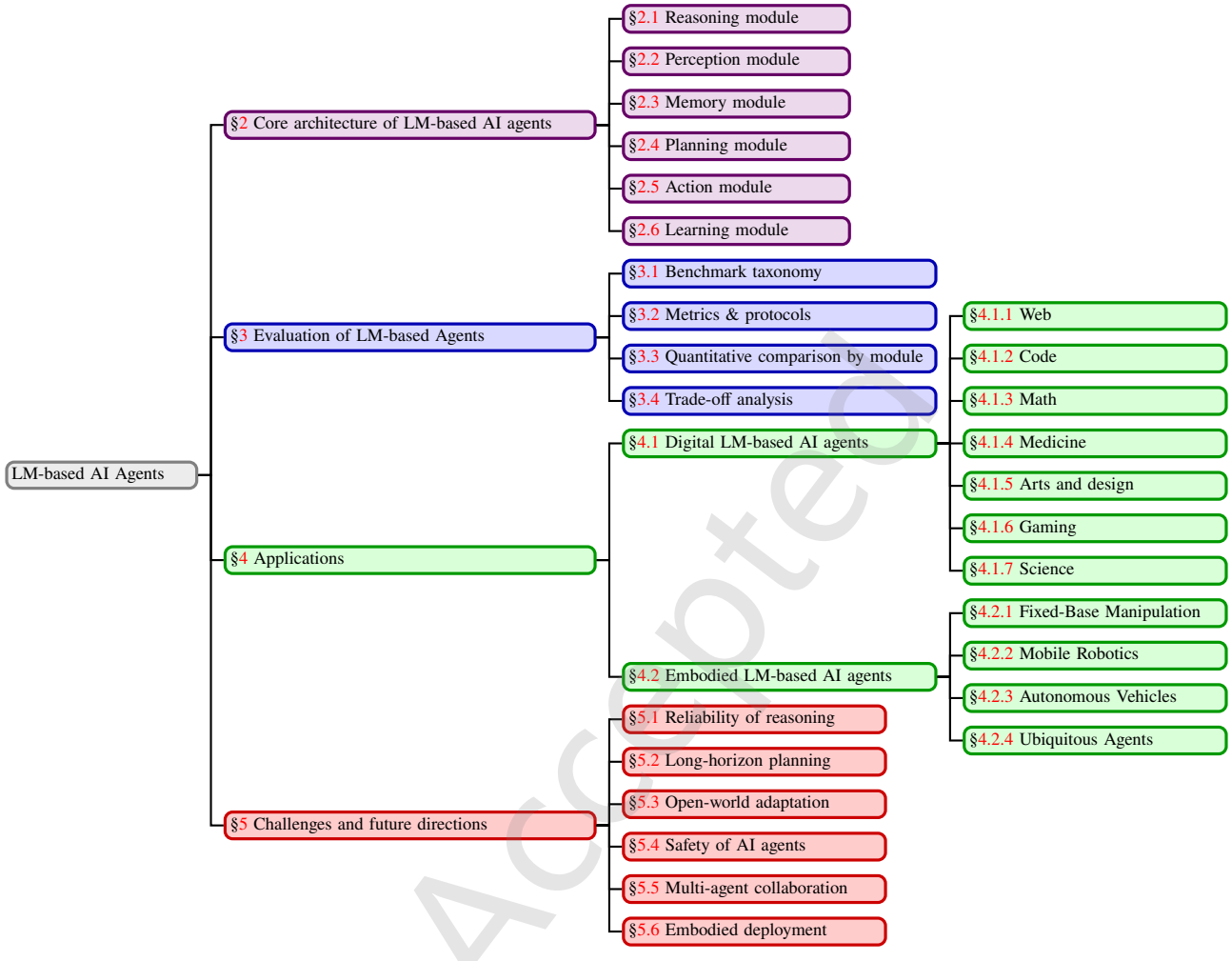


Fig. 1 The primary organizational structure of this survey.

Memory–Planning–Action skeleton and discuss applications in social science, natural science, and engineering. Reasoning and perception are not first-class modules, learning/reflection is omitted, and the framework is largely text-centric with no treatment of multimodal or embodied perception. Luo et al. [12] extend the same Profile–Memory–Planning–Action skeleton with Construction/Collaboration/Evolution axes and an emphasis on scientific-discovery applications. The survey adds a useful discussion of self-evolution but lacks a dedicated perception module and does not isolate reasoning as the orchestrating core. Cheng et al. [13] define agents through contextual cognition with four processes (encoding, perception, interaction, reasoning). The perspective is complementary to ours but is applied exclusively to digital agents (deep research, coding, GUI, and scientific domain). In addition, embodied sensory channels and autonomous-driving agents are not covered.

Taken together, these surveys leave four concrete gaps that our manuscript fills: (i) None treats reasoning as the architectural core that coordinates memory access, planning decomposition, action selection, and learning updates; (ii) None extends perception beyond the standard text/vision/audio triad to include spatial, proprioceptive, haptic, chemosensory, and neural channels; (iii) None promotes learning and reflection to a first-class module with explicit post-hoc improvement and capability-expansion sub-mechanisms; and (iv) None jointly covers GUI/web agents (as the digital extreme) and robotics/autonomous-driving agents (as the embodied extreme) within a single architectural framework. These four gaps collectively define the contribution of our survey. Table 1 summarizes the comparison. In this table, ✓ denotes full coverage, △ denotes partial coverage, and ✗ denotes no coverage. We define partial coverage as instances where existing literature briefly acknowledges a domain or explores only a subset of its capabilities, such as listing robotic applications without a systematic structural taxonomy. “Taxonomy” states each survey’s own presented framework. “Reas. core” denotes whether reasoning is treated as the coordinating core; “Percep.” de-

Table 1 Comparison With Prior LM-Agent Surveys.

Survey	Taxonomy	Reas. core	Percep.	Learn.	GUI/Web	Embodied	Year
Xi et al. [1]	Brain + Perception + Action	✗	△	✗	△	△	2025
Durante et al. [4]	Multimodal-interaction lens	✗	△	△	✗	△	2024
Wang et al. [11]	Profile–Memory–Plan–Action	✗	✗	△	✗	△	2025
Luo et al. [12]	Construction/Collab./Evolution	✗	✗	✓	△	△	2025
Cheng et al. [13]	Contextual Cognition (4 processes)	△	✗	△	✓	✗	2026
Ours	**6-module reasoning-centric**	✓	✓	✓	✓	✓	**2026**

notes whether perception extends beyond text/vision/audio to spatial, proprioceptive, haptic, chemosensory, and neural channels; "Learn." denotes whether learning/reflection is a first-class module; "GUI/Web" denotes systematic coverage of GUI and web agents; and "Embodied" denotes systematic coverage of robotics and autonomous-driving agents.

This survey addresses this critical need by providing a holistic and structured review of the LM-based AI agent landscape, as illustrated in Fig. 1. Our primary contributions are given as follows. First, we deconstruct the core architecture of modern agents in Section 2, proposing a unified conceptual framework that systematically analyzes the interplay among six foundational modules: Reasoning, perception, memory, planning, action, and learning. This provides a guideline for understanding and comparing disparate agent designs. Second, we systematically analyze the evaluation landscape in Section 3, providing a comprehensive taxonomy of benchmarks, metrics, and quantitative performance trade-offs across different modules. Third, we provide a panoramic view of the LM-based AI agents landscape by surveying its transformative impact across a broad spectrum of applications in Section 4, from digital domains such as software engineering [14] and scientific discovery [15] to embodied systems such as robotics [16] and autonomous vehicles [17]. This highlights the practical significance of the technology and inspires new use cases. Finally, in Section 5, we chart a course for future research by identifying the most critical challenges and promising directions, including robust reasoning, long-horizon planning, open-world adaptation, and agent safety. This serves as a roadmap for the community to tackle the next generation of problems in LM-based AI agents.

The remainder of this survey is organized as follows. Section 2 details the modular architecture of modern LM-based agents. Section 3 establishes a unified evaluation framework, detailing benchmarks, metrics, and module-specific performance. Section 4 showcases the diverse applications of these agents across both digital and embodied contexts. Section 5 discusses the primary obstacles facing the field and outlines promising directions for future research. Finally, Section 6 concludes this survey.

2 Core Architecture of LM-Based AI Agents

It is important to emphasize that this decomposition is conceptual and analytical: It provides a vocabulary for describing and comparing agent systems, not a prescription for their physical implementation. End-to-end architectures, in which multiple functions are realized through learned latent representations, are fully compatible with this framework. The architecture of an LM-based AI agent is best understood as a modular system orchestrated by an LM, which is the central cognitive core. As shown in Fig. 2, the entire operational loop of LM-based AI agents is driven by the interplay between the reasoning module and other modules, which are the perception, memory, planning, action, and learning modules.

The process begins when the perception module gathers raw data; however, it is the reasoning core that actively interprets this input, transforming ambiguous sensory information into a structured understanding of the world state. To contextualize this perception and inform its next steps, the agent reasons about what to store in or retrieve from its memory module, formulating effective queries for relevant experiences [18]. This context-aware reasoning is then applied to planning, where the LM decomposes high-level goals into actionable strategies [19]. When these strategies are executed, the reasoning core directs the action module by intelligently selecting appropriate tools and determining correct parameters for the task [9]. Finally, the entire process is subject to a meta-level feedback loop driven by the learning module [20]. Here, the agent reasons retrospectively about its performance, analyzing execution trajectories to identify errors and distill insights. These reflections are then used to determine how to update its memory, creating a virtuous cycle of continuous improvement. The following subsections then deconstruct the architecture of the agent, examining each of these foundational modules and their relationship with the central reasoning engine. Finally, we synthesize these individual components into a unified operational pipeline to illustrate the end-to-end execution flow and the principles of capability allocation.

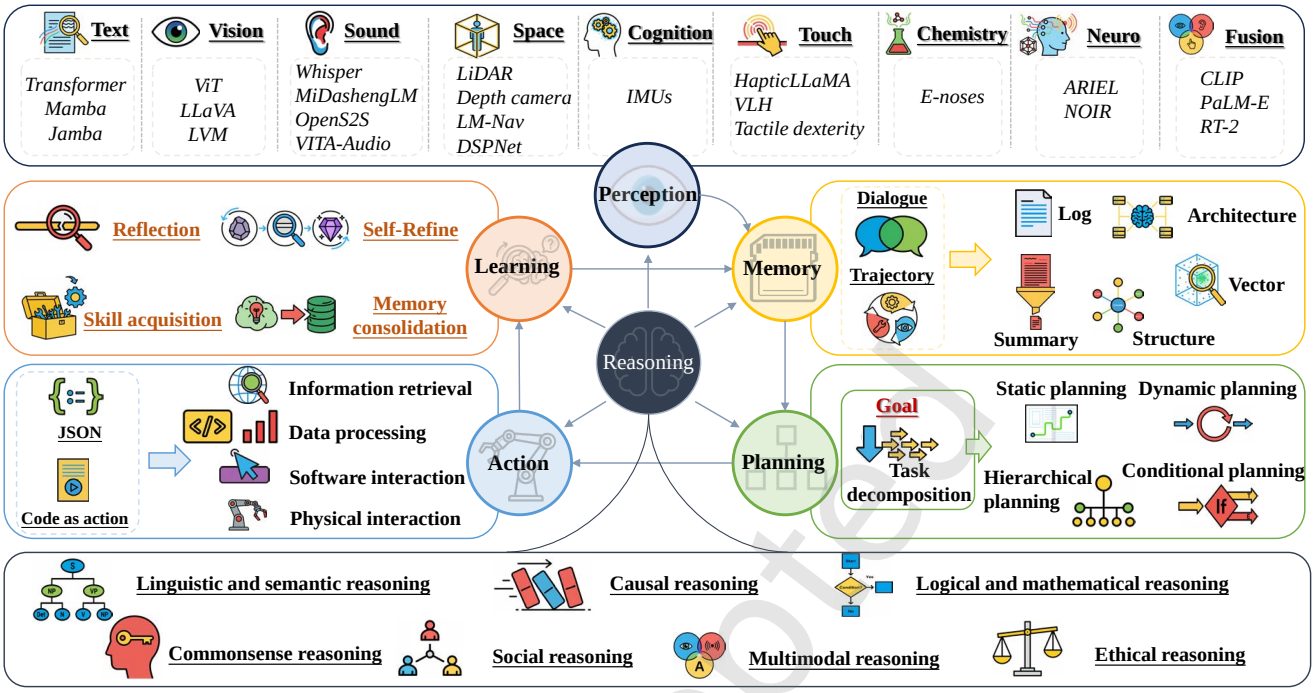


Fig. 2 The core architecture of an LM-based AI agent. The architecture is orchestrated by the central Reasoning module (dark blue). As detailed in the bottom panel, this central engine is powered by diverse intrinsic reasoning capabilities (e.g., linguistic, causal, logical, and multimodal reasoning). It interacts with five peripheral modules: (1) Perception (top): Integrates multimodal sensory inputs ranging from text and vision to neural and chemical signals. (2) Learning (top-left): Enables self-improvement through reflection, memory consolidation, and skill acquisition. (3) Memory (top-right): Stores and structures interaction history through logs, trajectories, and vector databases. (4) Action (bottom-left): Translates planned steps into executable operations, including application programming interface (API) calls, code execution, and physical interactions. (5) Planning (bottom-right): Decomposes complex goals into manageable tasks using static, dynamic, or hierarchical strategies.

2.1 Reasoning Module

This section introduces the reasoning module, which functions as the central cognitive engine of the LM-based AI agent [21]. We begin by categorizing the diverse reasoning capabilities endowed by LMs. Subsequently, we introduce key methodologies for enhancing these reasoning capabilities and explain how the reasoning module supports the operation of all other modules in the agent architecture.

2.1.1 Different Reasoning Capabilities

The LM endows the agent with a diverse set of intrinsic reasoning abilities acquired during pre-training [22]. Foundational to these is **linguistic and semantic reasoning**, the ability to understand complex grammar, context, and subtle semantics, which is crucial for understanding user instructions [23, 24]. In addition to literal understanding, agents employ **commonsense reasoning** to apply implicit knowledge of the physical and social world, thus making their behavior more rational and context-aware [25, 26]. For more structured tasks, agents exhibit **logical and mathematical reasoning**, enabling them to perform formal deductions and solve quantitative problems [27, 28]. A key frontier in this area is **causal reasoning**, the capacity to infer cause-and-effect relationships, which remains a developing yet critical capability for robust decision-making [29, 30]. In addition to these core competencies, advanced agents are increasingly demonstrating other specialized reasoning skills, such as **social, multimodal, and ethical reasoning** [31–33].

2.1.2 Impact of Reasoning on Each Module in LM-Based AI Agents

Reasoning is the core module of the LM-based AI agent and plays an important role in all modules. For the perception module, reasoning is applied to interpret and make sense of raw, often ambiguous, and multimodal inputs. It transforms sensory data into a structured understanding of the current state. With respect to the memory module, the reasoning module is crucial for both writing to and retrieving from memory. The agent must reason about which information is salient enough to be stored and formulate effective queries to retrieve the most relevant memories for the current context. For the planning module, the reasoning engine is invoked to evaluate logical dependencies and perform causal inference.

Rather than generating the temporal plan itself, reasoning provides the cognitive processing power required to validate the feasibility of sub-goals generated by the planner. For the action module, before execution, the agent determines which tool to use, what parameters to provide, and how to format its actions based on the plan and current state. For the learning module, after an action is performed, the agent determines the outcome to identify errors, attributes success or failure, and updates its future strategies.

2.1.3 Methods for Enhancing Reasoning Abilities

While powerful, the native reasoning abilities of LMs can still be brittle and unreliable. A significant body of research focuses on enhancing reasoning abilities through various methods, which can be broadly divided into inference scaling and embedding reasoning directly into an LM's weights [34].

Inference Scaling: Enhancing Reasoning at Inference Time Inference-scaling techniques aim to derive more robust reasoning from a fixed model by increasing the computational complexity of the inference process. A foundational strategy is chain-of-thought prompting, which directs the model to generate a series of intermediate reasoning steps before reaching an answer [35]. Building on this, self-consistency [36] improves the robustness of the reasoning by sampling multiple reasoning paths and taking a majority vote on the final answer. More complex exploratory structures, such as tree-of-thoughts [37], allow the agent to explore and self-evaluate different reasoning branches in a tree-like type. Furthermore, iterative methods such as Self-Refine [10] enable an agent to generate an initial solution and then repeatedly provide feedback to itself to refine and improve the output over multiple cycles.

Learning to Reason: Embedding Reasoning Into Model Weights This paradigm seeks to fundamentally enhance the intrinsic reasoning capabilities of an agent through specialized training methodologies. These techniques can be broadly categorized into three main approaches:

- Supervised reasoning learning: This approach involves fine-tuning a base model on high-quality datasets that contain explicit, step-by-step reasoning processes. Benchmarks such as GSM8K have detailed problem solutions [38], while subsequent efforts such as MetaMath [39] demonstrate the power of supervised fine-tuning on new, reasoning-focused datasets such as MetaMathQA to create powerful large-scale reasoning models.
- Outcome-supervised reinforcement learning (RL): In this approach, the model learns by interacting with an environment that provides a definitive reward based on the final outcome. The agent is rewarded for achieving a correct, verifiable result, thus encouraging it to explore and discover valid reasoning strategies. This paradigm is famously demonstrated by AlphaGo [40], which learns from the ultimate outcome of winning or losing a game, and is now applied in domains such as code generation, where models such as DeepSeek-Coder V2 are rewarded based on whether their code passes unit tests [41].
- Process-supervised RL: Representing the most advanced frontier, this approach provides rewards for each intermediate step of the reasoning process, not just the final outcome. Inspired by the success of RL from human feedback in aligning models with human preferences [42], this technique involves training a reward model on human comparisons of reasoning steps. This reward model then guides the agent to produce reasoning chains that are not only correct but also logical, coherent, and aligned with human thinking patterns [43].

2.2 Perception Module

In this section, we survey the diverse perception modules that endow agents with the ability to understand and interact with the world, ranging from processing textual and audiovisual signals to interpreting spatial, physical, and even neural inputs.

Textual Perception The foundation of an agent's textual perception lies in the Transformer architecture [44, 45], which revolutionizes natural language understanding through its self-attention mechanism. However, the quadratic computational complexity of attention motivates the investigation of more efficient alternatives for handling extremely long contexts. A leading method is the Mamba architecture, a type of state space model (SSM) that has linear-time complexity [46]. Furthermore, Jamba employs a hybrid Transformer-Mamba design to leverage the strengths of both architectures [47]. This trend marks a significant evolution in textual perception, particularly for agents requiring efficient processing of extensive conversational histories or large documents.

Visual Perception Visual perception equips an agent with the ability to "see" and interpret the world, forming a critical component for multimodal and embodied intelligence [48–51]. The process typically follows a two-stage architecture. First, a visual encoder, often a large vision model such as a vision Transformer (ViT) [52], processes raw pixel data to extract high-level semantic features into a set of visual embeddings. Second, these vision-only embeddings are translated into a

language-compatible space through a bridging mechanism. This is achieved either by aligning vision and text in a shared multimodal embedding space, pioneered by models such as CLIP [53], or by using a dedicated projector module to feed visual information into an LM, as demonstrated in modern vision-language models (VLMs) such as LLaVA [54]. This two-stage process allows the agent's cognitive core to seamlessly reason over visual information.

Auditory Perception Auditory perception equips an agent with the ability to process sound, a capability that is crucial for voice-enabled interactions and environmental awareness. At its core, this involves robust speech-to-text transcription, where models such as Whisper [55] convert spoken language into a machine-readable format. Beyond transcribing content, advanced auditory perception includes paralinguistic analysis, such as identifying speakers and recognizing emotional tones, which enables more socially aware interactions. Recent advancements include MiDashengLM, which offers strong audio understanding with efficient inference [56], OpenS2S, an empathetic large speech model [57], and VITA-Audio, which provides low-latency and efficient inference [58].

Spatial Perception Spatial perception enables an agent to construct a 3D model of its surroundings. Unlike standard cameras that capture 2D images, specialized sensors such as LiDAR [59] and depth cameras [60] provide direct geometric information, generating rich 3D point clouds or depth maps of the environment. These data enable the agent to understand the layout of a space, the location of objects, and to perform essential tasks such as navigation and obstacle avoidance. For instance, in robotic navigation, this spatial perception is fundamental for grounding high-level instructions within a physical map of the world, as exemplified by models such as LM-Nav [61] that leverage large pre-trained models for navigation. Furthermore, spatial perception is also crucial for robust scene understanding, as demonstrated by DSPNet [62] for 3D question answering.

Proprioceptive Perception Proprioceptive perception refers to an agent's ability to sense its internal state, such as the position and orientation of an LM-based robot. This is achieved primarily through sensors such as inertial measurement units (IMUs). IMUs measure acceleration and angular velocity, whereas joint encoders in robots report the precise angle of each joint. This self-awareness is the cornerstone of motor control, enabling an embodied agent to maintain balance, execute agile movements, and perform precise physical interactions with its environment. For example, legged robots rely critically on proprioceptive feedback to learn dynamic movement skills [63].

Haptic Perception Haptic perception [64] provides an agent with a sense of "touch," enabling it to perceive the world through physical contact and force feedback. The primary enablers are specialized sensors, such as force/torque sensors mounted on robot wrists and high-resolution tactile arrays on fingertips, which provide detailed data on interaction forces and contact pressures. This raw sensory information is increasingly interpreted by sophisticated multimodal systems. For example, models such as HapticLLaMA [65] can translate complex vibration signals into natural language descriptions, effectively captioning the sense of touch. In addition, foundation models such as VLH [66] integrate vision and language to synthesize haptic force and vibration feedback for tasks such as texture discrimination in aerial robotics. This advanced haptic perception, which enables the understanding of object properties such as texture and hardness, is essential for safe human-robot collaboration and for contact-based guidance in tasks such as peg-in-hole insertion in *tactile dexterity* [67].

Chemosensory Perception Chemosensory perception, leveraging technologies such as electronic noses (e-noses), aims to give agents a sense of "smell" for applications in industrial safety and quality control [68].

Neural Perception Perhaps the most transformative frontier is the use of brain-computer interfaces (BCIs) as a direct perception channel. This would allow an agent to perceive a user's cognitive state or intent directly from neural signals, paving the way for a new paradigm of truly seamless human-agent interaction. For example, the ARIEL framework enables an LM-based conversational agent to perceive a user's emotional state via electroencephalographic signals, allowing for more empathetic emotional support conversations [69]. In addition, NOIR explores how agents can interpret neural signals to control physical robots for everyday tasks [70].

Multimodal Fusion At the most advanced level, an agent's perception module must perform multimodal fusion [71, 72], which involves intelligently integrating information from its various sensory streams to form a single, coherent worldview. Key technical approaches include projecting different modalities into a shared embedding space, as pioneered by models such as CLIP [53], or using cross-attention mechanisms to allow one modality to query another [73]. The frontier of this

field lies in embodied multimodal models such as PaLM-E [74] and RT-2 [75], which directly embed a stream of real-time, multimodal sensor data into an LM, thus enabling seamless reasoning and action in the physical world.

2.3 Memory Module

The architecture of an agent's memory is crucial because it underpins its ability to maintain context, learn from past interactions, and perform complex reasoning over time [18]. The memory mechanism can be categorized along five key dimensions: The source of memory (what is remembered, §2.3.1), the temporal scope across which it is organized (short-term vs. long-term, §2.3.2), the lifecycle by which experiences are encoded, consolidated, and retrieved (§2.3.3), the storage format (how it is physically represented, §2.3.4), and the extension of memory through external knowledge (how Retrieval-Augmented Generation mitigates parametric limitations, §2.3.5).

2.3.1 *Sources of Memory*

Raw materials for an agent's memory are derived from its dialogues and execution trajectories:

- **Dialogues:** The complete history of conversations between the agent and users or other agents. Storing dialogues is essential for maintaining the conversational context, understanding user intent, and enabling personalized interactions [76–78].
- **Execution Trajectories:** The detailed log of the agent's internal reasoning and external actions, often structured as an "observation-thought-action" loop. This log is vital for complex task planning, learning from successes or failures, and ensuring procedural consistency [79–84].

2.3.2 *Temporal Scope of Memory*

Recent agent architectures have moved beyond flat storage toward hierarchically organized temporal tiers. H-MEM [85] constructs a four-level hierarchy organized by semantic abstraction depth, using positional index encoding for layer-aware retrieval and dynamic update mechanisms based on forgetting curves and feedback signals.

2.3.3 *Memory Lifecycle*

Agent memory operates through a structured lifecycle of encoding, consolidation, and retrieval. Mem0 [8] implements a two-stage pipeline: An LM first extracts important facts from interaction contexts, then performs adaptive consolidation by adding new memories, updating existing ones, or deleting outdated entries. HeLa-Mem [86] grounds the lifecycle in Hebbian learning principles.

2.3.4 *Storage Formats*

Once memory data are acquired, they must be stored in a way that is both efficient and accessible. The primary methods for storing memory can be summarized as follows:

- **Text/Logs:** The most straightforward approach, in which raw dialogue and execution trajectories are stored as plain text or structured logs [87, 88].
- **Summaries:** To manage the ever-growing volume of memory, LMs are used to create concise summaries of past interactions. This compresses information, reducing retrieval latency and computational costs [77–79].
- **Vector Databases:** The predominant method in modern agents. Information is converted into numerical vectors by embedding models and stored in a vector database. Retrieval is performed by searching for vectors with the highest semantic similarity to a given query [89–92].
- **Structured Knowledge:** Recent works have introduced specialized structured formats to improve memory organization. These include: An entity-centric, multimodal graph where nodes representing entities (e.g., people) store associated visual and auditory features [93]; a grounded world model that stores memories in a stable, personalized knowledge schema that maps information to the agent's environment [94]; and a hierarchical architecture that segregates memory into distinct tiers, such as "main" and "archival" stores, to manage information based on relevance and access frequency [95].
- **Sophisticated Memory Architectures:** These sophisticated memory architectures include: A centralized memory operating system, which provides a unified interface to manage all memory operations such as allocation and retrieval [96]; a self-controlled memory framework, where the LM itself learns to issue explicit commands to control the memory flow, such as deciding what to write or update [77]; and production-ready, scalable memory systems such as Mem0, which are designed with tiered architectures for efficient long-term storage and retrieval in real-world applications [8].

2.3.5 *Retrieval-Augmented Generation as a Memory Extension*

Retrieval-augmented generation (RAG) [16] serves as a pivotal technology for extending an agent's memory, transforming it from a static, parameter-bound knowledge store into a dynamic and extensible system. This paradigm fundamentally enhances the memory module by treating external document collections as a queryable, long-term memory repository. By leveraging vector-based semantic search, RAG allows an agent to retrieve relevant information from vast knowledge

bases in real-time, thus grounding its responses in factual, up-to-date, or domain-specific data and mitigating the inherent limitations of its internal parametric knowledge [97]. Some RAG frameworks implement sophisticated memory access patterns, such as context-aware or forward-looking active retrieval, ensuring that the information surfaced is precisely tailored to the agent's immediate task requirements [98]. By formalizing a mechanism for the agent to reflect on and selectively retrieve information, modern RAG architectures empower agents to generate responses that are not only more accurate but also more robust and verifiable [88, 99].

2.4 Planning Module

The planning module is responsible for converting high-level goals into structured, actionable procedures. Distinct from the reasoning module, which executes step-by-step logical inference and state evaluation, the planning module focuses on the structural and temporal organization of actions. It determines the sequence, hierarchy, and dependency logic of the steps required to achieve a desired outcome. It determines the sequence, hierarchy, and conditional logic of the steps required to achieve a desired outcome, thus charting the path from the current state to the goal state.

2.4.1 Core Function: Task Decomposition

The foundational act of planning is task decomposition. To process any non-trivial objective, the planner structures a high-level blueprint of achievable milestones. During this process, the planning module manages the structural graph (e.g., establishing prerequisite dependencies between tasks), while delegating the logical validation of each proposed subtask back to the reasoning engine. For example, the instruction "plan a research trip to a conference" is decomposed into sub-tasks like "find relevant conferences," "book travel and accommodation," and "prepare presentation materials." This decomposition creates the fundamental skeleton of the plan, upon which more detailed execution strategies are built [19, 100].

2.4.2 Planning Strategies

LM-based AI agents employ various strategies to generate and execute these plans.

Static Planning First, the agent generates a complete, end-to-end sequence of actions before execution. It synthesizes the entire plan based on its initial understanding of the goal and its world model. This "plan-first, act-later" methodology is prominent in recent prompting strategies such as Plan-and-Solve prompting, where the model first explicitly formulates a step-by-step plan and then executes that plan sequentially to derive a solution [19]. This approach is akin to creating a detailed blueprint before starting construction. It provides a clear and fully articulated path. However, when any step produces an unexpected outcome, the entire pre-computed plan may be invalidated, requiring a complete replan.

Dynamic Planning The dominant strategy for modern agents is dynamic or interleaved planning, which tightly couples planning with execution. Instead of formulating a complete plan, the agent typically plans only the next immediate action or a very short sequence. It then executes this action, observes the outcome from the environment, and uses this new information to inform the planning of the subsequent step. This iterative, reactive cycle is the cornerstone of the ReAct framework [88] and allows the agent to be highly adaptive.

Hierarchical Planning Building upon task decomposition, hierarchical planning organizes actions into a structure with multiple levels of abstraction. This strategy is critical for managing long-horizon tasks, as it allows an agent to operate at different levels of granularity, from high-level strategic goals to low-level executable actions. A prominent example is the Voyager agent, which explores the world of *Minecraft* by progressively building a hierarchical skill library. The LM planner sets a high-level goal, such as "iron mining," which is then decomposed into a sequence of simpler, previously acquired skills from its library [101]. This structure allows the agent to focus on a high-level strategy without becoming lost in low-level specifics, a principle also demonstrated in systems such as HuggingGPT, which constructs a dependency graph of tasks for different models [100].

Conditional Planning A more sophisticated approach involves creating plans that include conditional branches (e.g., if-then-else logic). Instead of a fixed sequence, the agent generates a policy that specifies different actions to take under different circumstances. For instance, a plan might include a step: "Attempt to book with Airline A; if seats are unavailable, then book with Airline B." This approach embeds contingency handling directly into the plan itself, thus making it more robust. A modern implementation of this is using LMs to generate executable code as policies, which naturally contain conditional logic to handle diverse scenarios [102].

2.5 Action Module

The action module translates the abstract intentions from the planning and reasoning modules into concrete operations that

alter the state of the external world.

2.5.1 *The Paradigm of Tool Use*

LMs possess vast knowledge but cannot interact with the world beyond generating text. To bridge this gap, agents are endowed with a set of tools, such as external functions, APIs, and physical actuators. This approach, famously demonstrated by systems such as Toolformer, allows an agent to augment its cognitive capabilities with practical, real-world functionalities [9]. These tools can be broadly categorized as follows:

- **Information Retrieval Tools:** These include functions that access external knowledge sources, such as a web search engine for real-time queries or a mechanism to query structured databases.
- **Data Processing Tools:** This category encompasses utilities for precise computation and analysis, ranging from simple calculators to powerful code interpreters for complex data manipulation and visualization.
- **Software Interaction Tools:** These are APIs that allow the agent to operate within digital environments, enabling it to send emails, book calendar events, or perform browser automation.
- **Physical Interaction Tools:** For embodied agents, these are interfaces to physical hardware that enable action in the real world, for instance, by issuing commands to move a robotic arm or instructing a drone to take a picture.

2.5.2 *Action Representation and Invocation*

The format in which an agent expresses its chosen action has evolved to become more robust and expressive. Initially, actions are represented as structured text, often a string with a specific keyword and parameters that require a custom parser to execute. The current standard is a machine-readable JavaScript Object Notation (JSON) object. This format aligns with the native tool calling or function calling capabilities integrated into modern foundation models, which dramatically improve the reliability and efficiency of tool invocation.

A more advanced and flexible paradigm is *code as action*, where the agent's output is an executable script, typically written in Python [102]. This representation is exceptionally powerful as it allows a single "action" to contain complex logic, including loops, conditionals, and variables. This enables the agent to perform multi-step operations and handle contingencies without repeatedly cycling through a high-level reasoning loop for every minor step.

2.6 Learning Module

The learning module transforms a static executor into a dynamic, adaptive system. Learning objectives span capability expansion, task adaptation, and experience-driven improvement, fed by diverse data sources and realized through parameter-based, memory-based, or feedback-based update mechanisms. The stability-plasticity dilemma and the online–offline trade-off are central challenges in this process. Together with post-hoc mechanisms such as Reflexion and long-term skill libraries, as in Voyager, these components enable agents to improve persistently over their operational lifetime.

2.6.1 *Taxonomy of Learning Objectives*

The learning objectives of LM-based agents can be systematically categorized along three complementary dimensions. Capability expansion refers to acquiring entirely new skills not present in the base model [9, 101]. Task adaptation involves specializing for domain-specific tasks through fine-tuning [103, 104]. Experience-driven improvement refines behavior based on execution feedback [20].

2.6.2 *Data Sources for Agent Learning*

Agent learning uses five main data sources. Human demonstrations provide expert-annotated trajectories [105], but they cost too much to scale easily. Self-generated trajectories from the agent's own actions act as learning signals [20, 101]. Environmental feedback directly guides the reasoning steps [88]. Synthetic data helps build early training datasets [39, 106]. Replay buffers save past experiences to prevent catastrophic forgetting [107].

2.6.3 *Taxonomy of Update Mechanisms*

Agent learning is realized through three families of update mechanisms. Parameter-based updates modify model weights, ranging from full fine-tuning to parameter-efficient alternatives [108] and parameter isolation [109]. Memory-based updates augment external memory. For example, skill libraries [101] write reusable modules into storage. Feedback-based updates leverage reward signals via reinforcement learning from human feedback (RLHF) with proximal policy optimization (PPO) [43] or direct preference optimization [110].

2.6.4 *Stability vs. Plasticity Trade-off*

The stability-plasticity dilemma, the tension between learning new knowledge and retaining prior capabilities, manifests as catastrophic forgetting in trained agents [111]. Three solution families address this in LM agents. **(i)** Gradient-constrained updates enforce orthogonal subspaces per task [112]; **(ii)** experience rehearsal dynamically synthesizes replay samples without storing raw data [113]; and **(iii)** modular parameter isolation [114].

2.6.5 Online vs. Offline Updates

Agent learning spans a spectrum from offline to online paradigms. Offline learning proceeds from pre-collected datasets [110], offering stability but being bounded by dataset coverage. Online learning interleaves environment interaction with updates [43, 115], enabling novel strategy discovery at the cost of sample inefficiency and instability. Hybrid approaches combine offline initialization with online refinement [116].

2.6.6 Post-Hoc Improvement Mechanisms

The primary mechanism for learning from mistakes is reflection, as formalized by the Reflexion framework [20]. When a task fails, the agent analyzes its action trajectory. Then, it acts as a critic to identify the root cause of the failure, generates a verbal self-critique, and stores this insight in its memory to guide subsequent attempts and avoid repeating the same error.

Unlike correcting failures, Self-Refine enables agents to iteratively improve the quality of their outputs [10]. This is achieved through a *Generate* $\rightarrow$ *Critique* $\rightarrow$ *Refine* loop, in which the agent first produces an initial draft of an output (e.g., a piece of code or a plan). It then provides feedback on its own work and uses that critique to generate an improved version. This iterative process allows for progressive quality enhancement without external feedback.

2.6.7 Capability Expansion and Long-Term Learning

Perhaps the most powerful form of learning is skill acquisition, in which an agent converts successful experiences into new, reusable capabilities. This is exemplified by the Voyager agent, which operates in an open-ended environment [101]. When Voyager successfully completes a novel, multi-step task, it abstracts the successful action sequence, writes it as a new function (a "skill"), and adds it to its growing skill library. This allows the agent to solve increasingly complex problems over time by composing previously learned skills, rather than reasoning from scratch.

This process of skill acquisition is tightly coupled with memory consolidation. The insights gained from reflection and the new skills curated are not ephemeral; they are synthesized and written to the agent's long-term memory. This ensures that learning remains persistent and contributes to a more robust and capable world model.

2.7 Operational Pipeline and Capability Allocation

This section unifies the six modules into a single, timed pipeline and gives the decision rule for capability allocation.

2.7.1 Operational Walk-Through

To trace how a single perceptual signal propagates through the architecture, we provide a concrete end-to-end walk-through using the request: *"Book a single non-stop flight from Beijing to Tokyo next Friday morning under $800."* The timed pipeline operates through the following sequence (t_0 to t_6), explicitly tracking active conceptual modules, datatypes, and computational costs:

- t_0 **Perception & Reasoning:** The LM directly takes the raw user request text as input and outputs a structured JSON intent (origin, destination, date, budget).
  Datatype: Text $\rightarrow$ JSON.
  Cost: **One LM inference call**.
- t_1 **Memory & Reasoning:** The system uses the JSON intent to query a vector database. The LM then processes retrieved historical records to synthesize a clean working context.
  Datatype: JSON $\rightarrow$ Text.
  Cost: **One LM inference call + Vector database lookup**.
- t_2 **Planning & Reasoning:** Taking the synthesized context as input, the LM outputs a step-by-step task execution plan.
  Datatype: Text $\rightarrow$ JSON.
  Cost: **One LM inference call**.
- t_3 **Action & Reasoning:** To prepare for tool execution, the LM processes the current sub-task context and outputs exact JSON parameters required to call the external flight search application programming interface (API).
  Datatype: Text $\rightarrow$ JSON.
  Cost: **One LM inference call**.
- t_4 **Action:** The system physically executes the external API call using generated JSON parameters and captures the raw flight data.
  Datatype: JSON $\rightarrow$ Text/JSON.
  Cost: API round-trip latency.
- t_5 **Perception & Reasoning Loop:** The raw API response is appended to the LM context as a text observation. The LM processes this observation text to decide the next step, repeating the tool formulation and execution cycle ($t_3 \rightarrow t_4$) until the plan is resolved.
  Datatype: Text $\rightarrow$ JSON/Text.
  Cost: **Multiple LM inference calls alternated with tool executions**.
- t_6 **Learning & Reasoning:** After the task concludes, a final LM call is invoked to review the entire execution trajectory and output a reusable code snippet or text critique, which is saved by the system.

Datatype: Text → Code/Text.
Cost: **One LM inference call**.

2.7.2 *Capability Allocation Rule*

The decision of whether a sub-skill is compiled into the LM weights or left outside as a tool hinges on three conceptual properties: Generality, volatility, and actuation. A sub-skill is typically absorbed into the LM's parametric weights (via pre-training or fine-tuning) if it exhibits high generality, low volatility (static knowledge), and requires no external actuation. Conversely, modern agents externalize a capability as a tool (e.g., an API call or a dynamically retrieved Markdown/code snippet in memory) if it exhibits low generality (highly specialized, domain-specific logic), high volatility (requires live environmental data, such as querying real-time flight prices), or demands actuation (executing real-world side effects, such as submitting a booking request).

3 Evaluation of LM-Based Agents

To rigorously compare the architectural designs presented in §2, this section establishes a unified evaluation framework. We first taxonomize the benchmark ecosystem (§3.1), then define metrics and experimental protocols (§3.2), present quantitative per-module performance comparisons (§3.3.1–3.3.6), and conclude with a cross-module trade-off analysis (§3.4).

3.1 Benchmark Taxonomy

We classify agent benchmarks into six categories spanning environments, capabilities, and safety.

3.1.1 *Web Navigation*

Web navigation benchmarks evaluate agents' ability to locate information, fill forms, and execute multi-step workflows across diverse websites. WebArena [117] and VisualWebArena [118] establish high-fidelity self-hosted environments with execution-based verification. Mind2Web [105] provides 2,350 human-annotated tasks across 137 websites in 31 domains, emphasizing cross-website generalization. BrowseComp [119] stress-tests deep browsing and cross-source information synthesis. Online-Mind2Web [120] extends evaluation to the live internet, trading reproducibility for realism.

3.1.2 *Code and Software Engineering*

Code benchmarks assess whether agents can understand, write, and debug code in real-world repositories [121]. SWE-bench [122] provides 2,294 real-world GitHub issues and pull requests for evaluating coding agents. BigCodeBench [123] targets code generation with diverse function calls and complex instructions. RExBench [124] assesses autonomous research-extension coding.

3.1.3 *Desktop and Operating System (OS) Interaction*

Desktop benchmarks evaluate agents' capacity to operate graphical interfaces and control operating systems. OSWorld [125] provides 369 real-computer tasks across Ubuntu desktop applications. OS-Harm [126] extends OSWorld with safety-critical tasks for evaluating agent robustness against harmful instructions. Windows Agent Arena [127] and Android-in-the-Wild [128] target platform-specific GUI control.

3.1.4 *Tool Use and API Calling*

Tool-use benchmarks measure how effectively agents select, compose, and execute external APIs to accomplish tasks. ToolBench [129] covers 16,464 real-world APIs with 274K instructions. NESTFUL [130] targets nested and multi-step API sequences. τ-bench [131] pairs tool-calling agents with simulated users for realistic interaction evaluation.

3.1.5 *Generalist Suites*

Generalist benchmarks span multiple domains to assess broad agent competence across reasoning, coding, and tool use. AgentBench [132] spans 8 environments. AgentGym [133] provides diverse tasks with programmatic reward signals for both training and evaluation. LifelongAgentBench [134] targets continual learning scenarios with 50+ sequential tasks. MAPS [135] offers a multilingual benchmark spanning reasoning, code, math, and security.

3.1.6 *Safety and Robustness*

Safety benchmarks probe agent resilience against adversarial attacks, harmful instructions, and out-of-distribution inputs [136]. AgentDojo [137] evaluates prompt injection in dynamic interactive environments. Agent-SafetyBench (ASB) [138] formalizes attack surfaces with 2,000 test cases and 349 environments.

3.2 Metrics and Experimental Protocols

Agent evaluation employs four metric categories. **Task Performance** measures goal achievement through Success Rate (fraction of tasks completed), Pass@k (probability of at least one success in k attempts, estimated via unbiased sampling), Progress Rate (normalized trajectory completion against an optimal reference), and Pass^k (consistency metric requiring all k attempts to succeed). **Output Quality** assesses correctness via standard natural language processing metrics and agent-specific measures such as tool selection accuracy and plan structural similarity. **Efficiency** quantifies resource consumption in steps/tokens, end-to-end latency, and cost per task. **Robustness** evaluates stability through consistency scores and adversarial success rates under perturbation. Depending on the specific task, standard experimental setups are

configured along five key dimensions: Shot setting (zero vs. few-shot), episode structure (single vs. multi-turn), success criteria (binary vs. graded), evaluation budget (API calls, tokens, wall-clock time), and reporting standards (confidence intervals, multiple random seeds). Furthermore, to ensure rigorous reproducibility, these setups demand fixed seeds, versioned environments, and open-source harnesses.

3.3 Quantitative Comparison by Module

To address the need for systematic performance assessment, we summarize representative quantitative findings for each architectural module, organized to mirror the six-module architecture presented in §2.

3.3.1 Reasoning

Reasoning quality is measured by task success rate as a function of reasoning depth and reasoning latency. On WebArena, frontier agents achieve a limited end-to-end task success rate of roughly 11% to 14.41%. The limited performance is primarily attributable to a lack of crucial capabilities such as active exploration and failure recovery [117]. On the decision-making benchmark, integrating sparse reasoning with interactive acting outperforms direct action generation by an absolute 10% in success rate [88].

3.3.2 Perception

Perception quality is evaluated through multimodal grounding accuracy on desktop and mobile environments. On OSWorld (369 real-computer tasks), the best model achieves only a 12.24% success rate, while pure-vision methods lag significantly behind [125]. Aria-UI (GPT-4o) reaches 44.8% on AndroidWorld when textual history is included, but drops to 39.7% without it, demonstrating that action history provides critical grounding context [139].

3.3.3 Memory

Memory retrieval quality is measured by memory retrieval accuracy and latency across increasing conversation history. On benchmarks like LoCoMo, MemOS [140] achieves the state-of-the-art overall score of 75.80, outperforming strong baselines. Furthermore, independent latency evaluations demonstrate that its innovative KV-based memory injection mechanism accelerates time to first token by up to 94.20% compared to standard prompt injection. H-MEM demonstrates that hierarchical memory with four semantic abstraction levels significantly improves memory retrieval accuracy (achieving an average F1 improvement of 14.98 points over baselines) while drastically reducing computational latency, processing retrievals about 5 times faster than traditional methods [85].

3.3.4 Planning

Planning capability is assessed by the planning success rate under compositional constraints. On TravelPlanner, the planning success rate is low: GPT-4 achieves only 0.6% final pass rate [141]. Flex-TravelPlanner [142] reveals that constraint order significantly affects planning success rate. Both GPT-4o and LLaMA 3.1 show higher success when global constraints are introduced after local constraints.

3.3.5 Action

Action quality centers on tool-calling success under varying action-space sizes. On MCPVerse, Claude-4-Sonnet attains 62.4% success under standard tool-calling conditions, but this drops to 44.2% when the action space expands to 147K tokens of tool schemas [143]. On ToolBench, when equipped with advanced reasoning strategies like DFSDT, frontier models and ToolLLaMA demonstrate strong multi-step API composition abilities, achieving pass rates well above 60% on complex multi-tool instructions [129].

3.3.6 Learning

Learning efficiency is measured by improvement curves over sequential tasks. On a sequence of 10 SuperNI tasks, models using the self-synthesized rehearsal mechanism achieve a significantly higher average ROUGE-L score of 63.23 compared to 17.33 for vanilla fine-tuning, while suffering massively less catastrophic forgetting (maintaining a backward transfer score of -1.56 compared to -53.64 for vanilla fine-tuning) [113]. Reflexion enables self-improvement through verbal reflection on task feedback, yielding substantial absolute gains across iterative trials (e.g., 22% on AlfWorld, 20% on HotPotQA, and 11% on HumanEval) [20].

3.4 Trade-off Analysis

For the Reasoning module, the choice of reasoning paradigm involves a trade-off between depth and efficiency: Detailed chain-of-thought improves accuracy but increases latency, while direct prompting is faster but less reliable on complex tasks. For the Perception module, multimodal perception trades grounding accuracy against processing cost: Visual inputs improve spatial reasoning but increase token count. For the Memory module, memory architecture choices trade retrieval quality against latency: Flat vector stores offer faster access but degrade over extended operation. For the Planning module, planning depth trades success rate against computational cost: Tree-search methods improve robustness but consume exponentially more tokens than linear decomposition strategies. For the Action module, tool selection strategies

require a careful balance between exploration and exploitation. While exhaustive schema loading maximizes capability, it inflates context length and slows inference. Conversely, retrieval-based filtering improves latency but risks missing relevant tools. For the Learning module, learning paradigm choices trade stability against plasticity: Online methods adapt faster to new tasks but exhibit higher variance, while offline approaches ensure consistency but are bounded by pre-collected data.

4 Applications

The conceptual framework of LM-based AI agents has rapidly translated into a diverse array of practical applications, fundamentally reshaping industries and augmenting human capabilities. These applications can be broadly categorized based on their operational domain: Digital agents, which inhabit virtual environments, and embodied agents, which interact with the physical world. Digital agents act as sophisticated cognitive assistants, automating complex workflows and enhancing creativity in domains ranging from software development and scientific discovery to medicine, gaming, and the creative arts. Conversely, embodied agents bridge the gap between computation and physical action, enabling autonomous systems such as fixed-base manipulators, mobile robots, autonomous vehicles, and ubiquitous agents to perceive, navigate, and manipulate their surroundings. This section surveys key applications within both categories, illustrating the transformative impact of agentic AI across this spectrum.

4.1 Digital LM-Based AI Agents

Digital agents operate in symbolic or visually rendered environments: Web pages, graphical user interfaces (GUIs), code repositories, mathematical formalisms, electronic health records, and digital media pipelines. Across these domains, we observe three macro-level trends: *Grounding modality* has diversified from pure textual Document Object Model (DOM) parsing to multimodal perception; *reasoning depth* has increased from single-step instruction following to multi-hop planning with self-reflection; and *tool integration* has evolved from hard-coded API calls to dynamic tool discovery and composition.

4.1.1 Web

We organize web agents by their *tool usage pattern*, specifically how they observe and interact with the browser environment.

Code-as-Action. Agents synthesize executable code to control the browser, combining language model reasoning with programmatic web interaction. WebAgent [144] pioneers this by generating Python programs via Selenium WebDriver, using an HTML-T5 encoder to parse and summarize DOMs and a Flan-U-PaLM decoder for grounded code synthesis. Related agents include WebRL [145] and WebLINX [146]. The key lesson is that code synthesis enables flexible browser automation, yet reliance on structured DOM representations limits robustness on visually complex or dynamic interfaces.

DOM API. Agents parse structured browser DOMs as programmatic APIs, ranking candidate elements and predicting symbolic actions. Mind2Web [105] exemplifies this through a two-stage pipeline: A DeBERTa cross-encoder ranks candidate DOM elements, and an LM predicts the action conditioned on ranked elements. Related agents include WebArena [117], VisualWebArena [118], AutoWebGLM [147], and SeeClick [148]. The key lesson is that treating the browser DOM as a structured API surface enables strong performance on standard websites, yet dynamic content in JavaScript-heavy single-page applications renders this approach fragile.

Visual Grounding. Agents perceive the interface through screenshots and generate actions via visual grounding, eliminating dependency on structured markup. WebVoyager [149] feeds full-page screenshots into a vision-language model (VLM) and generates element-level actions via Set-of-Mark (SoM) visual grounding. Related visual-grounding agents include AppAgent [150], Mobile-Agent-V2 [151], Ferret-UI [152], Ferret-UI 2 [153], Aguvis [154], ShowUI [155], and LASER [156]. Surveys by OS Agents [157] and Agentic Web [158] provide comprehensive overviews, with OpenWebAgent [159] offering an open toolkit. The key lesson is that visual grounding enables cross-platform generalization to any visual interface, yet the approach incurs high compute costs and struggles with small-target localization.

4.1.2 Code Generation and Debugging

Code agents generate and manipulate program syntax. We group them by tool usage pattern.

Compiler and Test Runner. The dominant pattern leverages compiler feedback and test execution to close a generate-repair loop. SWE-Agent [14] designs a specialized agent-computer interface (ACI) that exposes file navigation, text search, and line editing tools to the LM, enabling autonomous browsing of code repositories, editing files, and running tests. This pattern requires well-structured test suites and is primarily limited to supported languages. Foundational

code models enabling this pattern include Code Llama [160], Qwen2.5-Coder [161], and DeepSeek-Coder-V2 [104]. Related tool-augmented approaches include CodeAgent [162] and OpenCodeInterpreter [163]. The key lesson is that leveraging compiler feedback and test execution creates a reliable generate-repair loop for code generation, yet the approach requires well-structured test suites and remains limited to supported languages.

Multi-Agent Collaboration. A second pattern distributes software engineering tasks across specialized agents [164]. AgentCoder [165] assigns the distinct roles of programmer, test-designer, and test-executor to iteratively refine code through structured feedback protocols. This decomposition handles complex programming tasks but introduces coordination overhead. The key lesson is that distributing software engineering tasks across specialized agents with distinct roles enables handling complex programming tasks through structured feedback, yet the coordination overhead increases with task complexity.

4.1.3 Math

Mathematical reasoning agents are grouped by their evaluation style, specifically by how they verify correctness.

Formal Proof Verification. These agents integrate LMs with theorem provers that provide machine-checkable correctness guarantees. DeepSeek-Prover [166] advances this paradigm through large-scale synthetic data generation and RL from proof-verification feedback, achieving outstanding results. The approach is confined to well-formalized mathematical domains. Related agents include AlphaGeometry [15] and ProverAgent [167]. The key lesson is that integrating LMs with theorem provers via synthetic data and RL feedback provides machine-checkable correctness guarantees, yet the approach is confined to well-formalized mathematical domains.

Tool-Assisted Numerical Verification. Operating in natural language, these agents use tools (Python interpreters, symbolic solvers) to verify intermediate results empirically. ToRA [168] interleaves analytical reasoning steps with Python code execution. This pattern covers broad mathematical territory but sacrifices formal correctness guarantees. Supporting models include Llemma [169], DeepSeekMath [170], Qwen2.5-Math [171], MetaMath [39], OpenMathInstruct-2 [172], and Math-Shepherd [173]. The key lesson is that interleaving reasoning steps with Python code execution broadens the scope of verifiable mathematical problems, yet the approach sacrifices formal correctness guarantees inherent in theorem-prover-based methods.

4.1.4 Medicine

Medical agents are grouped by their level of autonomy, measured by the degree of human oversight in the decision loop.

Assistive. At this level, the agent augments human clinicians, but the final decision remains with the physician. Med-Gemini [103] provides broad multimodal medical understanding across text, imaging, and structured records, with a knowledge-retrieval mechanism that grounds diagnostic suggestions in evidence-based medical literature. Related works include UltraMedical [174], TopoPharmDTI [73], VILA-M3 [175], Medical GraphRAG [176], MedCalc-Bench [177], LLM4DEU [178], FACTOR [179], and medical text representation frameworks [180]. The key lesson is that grounding diagnostic suggestions in evidence-based literature with a physician-in-the-loop ensures safety and interpretability, yet the resulting cognitive overhead bounds clinical utility by the level of trust clinicians place in the system.

Autonomous Collaboration. These agents autonomously synthesize information from multiple sources through multi-agent collaboration. MDAgents [181] simulates a multi-disciplinary team in which specialist agents exchange findings and converge on a diagnostic consensus without human intervention in the collaboration loop. The key lesson is that simulating a multi-disciplinary team where specialist agents autonomously exchange findings can handle complex clinical scenarios through diagnostic consensus, yet the pattern faces significant regulatory hurdles and challenges of distributed accountability.

4.1.5 Arts and Design

Creative applications span visual arts, music, and cross-modal composition. We group them by environment type: The modality and structure of the creative environment.

Virtual Social World. Operating in open-ended simulated societies, these agents model human behavior to produce emergent social dynamics. Generative Agents [87] simulate believable individual and collective behavior in a virtual town through a memory stream, reflection mechanism, and planning system, enabling emergent events such as relationship formation and information diffusion. The key lesson is that a memory stream combined with reflection and planning

enables believable simulation of individual and collective social behavior, yet evaluation remains inherently subjective, and controllability over long-horizon narratives is limited.

Visual and Spatial Media. These agents generate and edit visual content spanning 2D images, 3D scenes, and video. GenArtist [182] unifies image generation and editing under a single multimodal LM agent, interpreting free-form user requests and orchestrating diffusion models to produce or modify visual content. Related agents include ChatGen [183], MM-StoryAgent [184], MagicQuill [185], SceneWiz3D [186], TC4D [187], V-Stylist [188], and DiffSensei [189]. The key lesson is that unifying art generation and editing under a single multimodal LM agent enables strong creative potential from free-form requests, yet fine-grained control over stylistic consistency remains a challenge.

Music and Audio Composition. These agents create and manipulate musical and auditory content through tool-coordinated generation. MusicAgent [190] serves as an AI agent for music understanding and generation, leveraging large language models to interpret musical concepts and orchestrate specialized audio models for composition, arrangement, and style transfer. Related audio agents include WeaveMuse [191], Loop Copilot [192], FilmComposer [193], and AudioGenie [194]. The key lesson is that using LMs to interpret musical concepts and orchestrate specialized audio models enables flexible composition and arrangement, yet musical quality assessment remains largely subjective and dependent on human aesthetic judgment.

4.1.6 *Gaming and Interactive Entertainment*

Gaming agents are grouped by game environment type: The structure and openness of the game world they inhabit.

Open-World Sandbox. In open-ended sandbox environments such as Minecraft, agents explore, acquire skills, and complete tasks without predefined objectives. Voyager [101] explores the open world of Minecraft through an autonomous skill library: When it discovers a new capability, it abstracts the successful code as a reusable skill, composes skills to solve increasingly complex tasks, and stores them for future use. Related works include GITM [195], ODYSSEY [196], Auto MC-Reward [197], MP5 [198], and MindAgent [199]. The key lesson is that abstracting successful action sequences as reusable code skills stored in an autonomous library enables lifelong learning through composition, yet the approach fundamentally depends on a stable, long-horizon environment.

Strategy Games. In competitive strategy game environments such as *Diplomacy* and *StarCraft*, agents face adversarial opponents and must negotiate, plan, and adapt in real time. Cicero [200] achieves human-level play in Diplomacy by combining a strategic reasoning engine with a dialogue model, tightly coupling language and strategy to negotiate, deceive, and form alliances. Related strategy agents include Richelieu [201], SwarmBrain [202], and LangBirds [203]. The key lesson is that tightly coupling a strategic reasoning engine with a dialogue model enables powerful performance, yet performance is tightly coupled to game-specific mechanics with poor cross-game transfer.

Text-Adventure Worlds. In text-driven narrative environments, agents generate interactive game worlds and stories from natural language, creating playable experiences through linguistic descriptions. For instance, Word2World [204] generates interactive game worlds and stories from natural language descriptions, creating playable environments through large language models. Related text-based agents include AWM-BART [205]. The key lesson is that generating interactive game worlds directly from natural language descriptions enables highly flexible creative content, yet narrative consistency degrades as the generated world scales in complexity.

4.1.7 *Science*

Scientific discovery represents one of the most transformative frontiers for LM-based AI agents, where autonomous systems are accelerating research across chemistry, physics, biology, and materials science. We group them by discovery paradigm.

Autonomous Hypothesis Generation and Verification. Agents autonomously formulate hypotheses, design experiments, and iteratively refine theories based on empirical feedback. Coscientist [206] leverages GPT-4 with tool integration to autonomously plan and execute chemical experiments. Related autonomous discovery agents include AlphaEvolve [207] and ChemCrow [208]. The key lesson is that autonomously formulating hypotheses and executing experiments via domain-specific tool integration accelerates scientific discovery, yet the approach relies heavily on expert-engineered tools and struggles with unformalized domain tacit knowledge. Fig. 3 illustrates an agent-driven defect detection pipeline for Ni-Co alloy plates.

Simulation-Driven Discovery. Agents orchestrate multi-code simulation workflows end-to-end. TopoMAS [209] coordinates specialized agents through a core orchestrator,

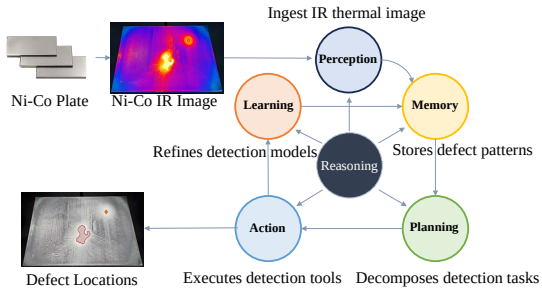


Fig. 3 LM-based AI Agent for Ni-Co plate defect detection.

autonomously integrating computational outcomes into a dynamic knowledge graph for iterative refinement. The key lesson is that LM-based agents accelerate scientific discovery by orchestrating multi-code workflows and integrating predictive modeling with experimental validation, yet reliability depends on domain-specific tacit knowledge and simulation fidelity [210].

4.2 Embodied LM-Based AI Agents

Embodied LM-based AI agents represent a critical frontier in artificial intelligence, enabling agents to perceive, reason, and act within the physical world [211–213]. Embodied agents are categorized based on hardware embodiment: *Physical platform morphology and locomotion capability*, as this determines the sensor suite, action space, and safety constraints. We examine four categories: Fixed-base manipulation, mobile robotics, autonomous vehicles, and ubiquitous agents.

4.2.1 Fixed-Base Manipulation

Fixed-base manipulation systems operate stationary articulated arms in structured environments [214]. We classify by *tool usage pattern*: The mechanism through which LM outputs are converted into physical actions.

Semantic Abstractions and Composable Programs as Tools. In this pattern, the LM performs high-level semantic reasoning and outputs a structured intermediate representation, either a spatial abstraction or an executable program, which a specialized downstream system then translates into low-level physical actions. VoxPoser [215] exemplifies this approach: It prompts an LM to generate code that instantiates composable 3D value maps encoding spatial constraints, which are fed into a motion planner for trajectory optimization. Related works include PhysVLM [216], ProgPrompt [217], LM-Planner [218], SELP [219], and REFLECT [220]. The key lesson is that separating semantic planning from physical execution via structured intermediate representations enables interpretability and zero-shot generalization, yet the approach remains sensitive to perception noise and bounded by the coverage of pre-engineered primitives and planners.

Code Synthesis for Reward and Policy. This pattern uses LMs to directly synthesize manipulation policies as executable code or reward functions. Eureka [221] leverages LMs to perform human-level reward design via coding, iteratively refining reward specifications based on training outcomes. By automating reward engineering, it trains policies for diverse manipulation skills without human-designed rewards. The key lesson is that automating reward engineering via LM coding eliminates manual reward design, but the iterative RL loop incurs substantial computational cost and risks unsafe exploration behaviors.

End-to-End VLA as Unified Tool. In this pattern, the LM treats a pre-trained VLA model as a monolithic tool that directly maps visual observations and natural-language instructions to low-level motor commands. This paradigm collapses perception, reasoning, and control into a single neural function, trading modularity for closed-loop latency and cross-task generalization. RT-2 [75] exemplifies this approach by co-training on web-scale vision-language data and robotic trajectories, outputting robot actions as language tokens and enabling zero-shot generalization to novel objects and instructions. Related works span RT-1 [222] and π_0 [223]. The key lesson is that end-to-end VLA models serve as powerful unified tools for closed-loop control and generalize across objects, instructions, and robot morphologies, but they require large trajectory datasets for pre-training, sacrifice the interpretability and debuggability of modular pipelines, and are sensitive to perceptual domain shift at deployment time.

4.2.2 Mobile Robotics

Mobile robotic agents navigate spatially extended environments while maintaining coherent spatial representations. We classify by *environment type*, as the structural properties shape perception requirements and spatial reasoning complexity.

Indoor Structured Environments. Indoor environments feature structured layouts and semantic regions that LM agents leverage for landmark-based navigation. LM-Nav [61] composes three pre-trained models: A language model for goal parsing, a VLM for landmark recognition, and a visual navigation model for path following, thereby enabling zero-shot navigation from natural language instructions. Related works include TidyBot [224] and OK-Robot [225]. The key lesson is that modular composition of pre-trained models

achieves zero-shot language-conditioned navigation without task-specific training, yet brittle inter-module alignment can propagate errors across the pipeline.

Outdoor Unstructured Environments. Outdoor environments introduce uneven terrain, dynamic obstacles, and the absence of structured cues. NaVILA [226] presents a vision-language-action model for legged robot navigation, generating whole-body control commands that account for terrain geometry and dynamic stability. Related approaches include SPINE [227], CoNVOI [228], and VLFM [229]. The key lesson is that end-to-end VLA models can directly output low-level control commands for rough-terrain locomotion, but the sim-to-real gap remains a fundamental barrier to generalizing across diverse natural terrain.

Multi-Robot Collaborative Environments. Multi-robot collaboration [230] requires distributed coordination through natural language. RoCo [231] demonstrates dialectic multi-robot collaboration, where robots engage in natural language dialogue to negotiate task assignments and coordinate joint actions. Related work includes Smart-LLM [232]. The key lesson is that multi-turn natural language dialogue enables decentralized task coordination without a central planner, yet the negotiation latency and inconsistency under local failures limit real-time applicability.

4.2.3 *Autonomous Vehicles*

Autonomous vehicles represent the most safety-critical domain, operating at high speeds with split-second decision requirements [233, 234]. We classify by *tool usage pattern*: The architectural relationship between the LM and the vehicle's control system.

End-to-End VLA Driving. This pattern trains vision-language-action models to map sensor observations directly to driving commands through unified neural architectures. LMDrive [17] is a closed-loop end-to-end driving framework that processes multi-camera video and natural language instructions through a transformer-based VLA architecture, re-evaluating control decisions at each timestep. Related works include EMMA [235], OpenEMMA [236], AutoVLA [237], ORION [238], ReCogDrive [239], WiseAD [240], and AlphaDrive [241]. The key lesson is that unified VLA architectures can seamlessly map visual observations and language instructions to driving commands, but decision opacity fundamentally hinders safety verification and regulatory approval.

Generative Driving Policies. This paradigm leverages generative models to produce driving policies and simulate future scenarios for planning. Gen-Drive [242] enhances diffusion generative policies with reward modeling and RL fine-tuning, training a diffusion model to generate future trajectory rollouts and a learned reward model to score trajectories. Related work includes Sce2DriveX [243] for scene-to-drive learning with multimodal language models. Datasets enabling these approaches include OmniDrive [244], which provides holistic vision-language data with counterfactual reasoning for autonomous driving. The key lesson is that generative models enable proactive planning by simulating future scene rollouts, but the computational cost of online generation and the risk of divergence from real-world physics limit practical deployment.

4.2.4 *Ubiquitous Agents*

Ubiquitous agents embed LMs in everyday environments, such as smart homes, wearable devices, and distributed sensor networks. We classify by environment type.

Smart-Home Internet of Things (IoT) Environments. Smart-home agents orchestrate networks of IoT devices to fulfill user requests in natural language. SAGE [245] dynamically constructs LM prompt trees to translate high-level goals into grounded device action sequences through tool orchestration, with test harnesses for validation and failure recovery. Related works include HomeBench [246], LLMind [247], and AutoIoT [248]. The key lesson is that a structured knowledge graph of device capabilities enables grounded execution of high-level language goals, but reliance on pre-defined ontologies and fragmented IoT protocols demands extensive manual integration.

Augmented Reality/Virtual Reality (AR/VR) and Wearable Environments. Agents in AR/VR process multimodal inputs in real time while respecting user cognitive load. GazePointAR [249] fuses egocentric visual observations, gaze direction, and spoken pronouns to resolve referential ambiguity in user commands. Related works include AgentAR [250]. The key lesson is that fusing egocentric vision with gaze and language resolves referential ambiguity for wearable assistance, but severe resource constraints of head-mounted devices limit model size and inference throughput.

Aerial and Distributed Sensing Environments. Aerial platforms face challenges of decentralized coordination and energy-constrained operation. SwarmGPT [251] demonstrates

Table 2 Summary of Digital LM-Based AI Agents

Task Domain	Task Category	Key Design Idea	Key Limitation
Web	Code-as-Action	Synthesize executable code (e.g., Python via Selenium WebDriver) to control the browser programmatically.	Reliance on structured DOM representations limits robustness on visually complex or dynamic interfaces.
	DOM API	Parse HTML DOM as structured API; rank candidate elements, then predict actions via LM.	Fragile on JavaScript-heavy single-page applications.
	Visual Grounding	Use VLM over full-page screenshots with Set-of-Mark (SoM) for element-level action generation.	High compute cost; struggles with small-target localization.
Code	Compiler and Test Runner	Close generate-repair loop via compiler feedback and test execution through an Agent-Computer Interface.	Requires well-structured test suites; limited to supported languages.
	Multi-Agent Collaboration	Distribute tasks across specialized agents (programmer, test-designer, test-executor) with iterative feedback.	Coordination overhead increases with task complexity.
Math	Formal Proof Verification	Integrate LMs with theorem provers via synthetic data generation and RL from proof-verification feedback.	Confined to well-formalized mathematical domains.
	Tool-Assisted Numerical Verification	Interleave reasoning steps with Python code execution for empirical verification of intermediate results.	Sacrifices formal correctness guarantees.
Medicine	Assistive	Augment clinical decision-making through evidence-based grounding, multimodal diagnosis, and specialized biomedical tools.	Physician cognitive overhead bounds clinical utility by trust level.
	Autonomous Collaboration	Simulate a multi-disciplinary team where specialist agents autonomously exchange findings to converge on a consensus.	Regulatory hurdles and distributed accountability.
Arts and Design	Virtual Social World	Memory stream + reflection + planning to simulate believable individual and collective behavior.	Subjective evaluation; limited long-horizon controllability.
	Visual and Spatial Media	Unified multimodal LM agent interprets free-form requests and orchestrates diffusion models for image/video/3D.	Limited fine-grained control over stylistic consistency.
	Music and Audio Composition	LM interprets musical concepts and orchestrates specialized audio models for composition and arrangement.	Musical quality assessment is inherently subjective.
Gaming	Open-World Sandbox	Abstract successful action sequences as reusable code skills stored in an autonomous library for lifelong composition-based learning.	Depends on stable, long-horizon environments.
	Strategy Games	Couple strategic reasoning with language models for game-specific decision making, from diplomacy negotiation to real-time tactics and physics-based puzzle reasoning.	Tightly coupled to game-specific mechanics; poor cross-game transfer.
	Text-Adventure Worlds	Generate interactive game worlds and stories directly from natural language descriptions.	Narrative consistency degrades at scale.
Science	Hypothesis Gen. & Verification	Autonomously formulate hypotheses, plan, and execute experiments via domain-specific tool integration.	Relies heavily on expert-engineered tools; struggles with unformalized domain tacit knowledge.
	Simulation-Driven Discovery	Orchestrate multi-code simulation workflows end-to-end via dynamic knowledge graph integration.	Reliability depends on domain-specific tacit knowledge and physical fidelity of underlying simulators.

drone swarm choreography, where an LM generates spatial formation patterns and collision-avoidance constraints from natural language descriptions. Related works include UAV-VLA [252], FlockGPT [253], and AerialAgent [254]. The key lesson is that natural language provides an intuitive interface for specifying complex swarm formations, but centralized planning introduces a single point of failure, and LM-generated flight plans require formal safety verification before deployment.

To facilitate comparison across the diverse agent architectures surveyed in this section, Table 2 and Table 3 compactly summarize the key design ideas and limitations of each task category.

5 Challenges and Future Directions

Despite the rapid advancement in LM-based AI agents, the path toward creating truly autonomous, robust, and general agents is fraught with significant challenges. This section outlines the key obstacles that the field must overcome and highlights promising directions for future research.

5.1 Robustness and Reliability of Reasoning

A fundamental challenge is the robustness and reliability of an LM's reasoning. These models remain susceptible to factual hallucinations, logical fallacies, and a superficial grasp of causality [255]. This unreliability presents a significant barrier to their deployment in high-stakes applications such as medicine and finance. Future work must therefore focus on enhancing the robustness of reasoning. Promising directions include the integration of neuro-symbolic methods [256],

Table 3 Summary of Embodied LM-Based AI Agents

Task Domain	Task Category	Key Design Idea	Key Limitation
Fixed-Base Manipulation	Semantic Abstractions and Composable Programs as Tools	LM performs high-level semantic reasoning to output structured intermediate representations (spatial abstractions or executable programs) for downstream execution.	Sensitive to perception noise; bounded by coverage of pre-engineered primitives and planners.
	Code Synthesis for Reward and Policy	Automate reward engineering via LM coding to synthesize manipulation policies without human-designed rewards.	Iterative RL loop incurs substantial computational cost and risks unsafe exploration behaviors.
	End-to-End VLA as Unified Tool	Collapse perception, reasoning, and control into a single neural function mapping visual observations and language directly to motor commands.	Requires large trajectory datasets; sacrifices interpretability and debuggability; sensitive to perceptual domain shift.
Mobile Robotics	Indoor Structured Environments	Compose pre-trained LM, VLM, and visual navigator for zero-shot language-conditioned navigation.	Brittle inter-module alignment propagates errors across the pipeline.
	Outdoor Unstructured Environments	VLA model outputs low-level joint commands for legged locomotion over rough terrain from language.	Sim-to-real gap for diverse natural terrain.
	Multi-Robot Collaborative Environments	Multi-turn natural language dialogue for decentralized task negotiation and joint action coordination.	Negotiation latency; plan consistency under local failures.
Autonomous Vehicles	End-to-End VLA Driving	Closed-loop end-to-end driving framework processes multi-camera video and natural language instructions through a transformer-based VLA architecture, re-evaluating control decisions at each timestep.	Decision opacity fundamentally hinders safety verification and regulatory approval.
	Generative Driving Policies	Diffusion generative models with reward modeling produce driving policies and simulate future trajectory rollouts for planning.	High compute cost of online generation; risk of divergence from real-world physics.
Ubiquitous Agents	Smart-Home IoT Environments	Dynamically construct LM prompt trees and tool orchestration to translate language goals to grounded device commands.	Reliance on fragmented IoT protocols and pre-defined ontologies demands extensive manual integration.
	AR/VR and Wearable Environments	Fuse egocentric vision, gaze, and speech to resolve referential ambiguity in user commands.	Severe resource constraints of head-mounted devices limit model size and inference throughput.
	Aerial and Distributed Sensing Environments	LM generates parameterized swarm control policies and spatial formations from natural language descriptions.	Centralized planning introduces a single point of failure; requires formal safety verification.

which combine neural pattern recognition with the formal guarantees of symbolic logic, as well as the development of sophisticated self-correction mechanisms [257] that enable agents to rigorously validate their reasoning chain prior to acting. For instance, on τ-bench, Claude-4-Sonnet's tool-calling success drops sharply when the action space expands to 147K tokens of tool schemas [143], and frontier agents on WebArena plateau at limited end-to-end success, with the gap partly attributable to the poor reasoning ability [117]. These limitations reflect three recurring failure modes. **(i)** *Brittle tool selection*: Agents rely on surface-level schema matching rather than semantic understanding, causing accuracy to collapse when tool descriptions are unfamiliar or oversized. **(ii)** *Inconsistent planning*: Chain-of-thought improves accuracy, but longer reasoning chains accumulate intermediate-step errors without proportional reliability gains, and tree-search methods consume exponentially more tokens. **(iii)** *Fragile cross-modal reasoning*: Perceptual noise in embodied settings propagates through the reasoning module to produce incorrect actions, as seen in VoxPoser's sensitivity to 3D reconstruction errors. These patterns share a common root: The absence of explicit uncertainty quantification and pre-execution verification. We identify the following open questions for the community:

- How can agents quantify uncertainty in intermediate reasoning steps and request clarification when confidence falls below a threshold?
- What mechanisms can verify tool-parameter compatibility before execution, rather than relying solely on post-hoc error recovery?
- Can process-supervised reward models be extended to rate reasoning chains in open-ended agent tasks where verifiable outcomes are scarce?

5.2 Long-Horizon Planning and Strategic Foresight

Current agent architectures, particularly those based on reactive frameworks such as ReAct, excel at short-term, step-by-step tasks. However, they often struggle with long-horizon planning that requires strategic foresight and the ability to navigate complex dependency chains [258]. A purely reactive agent may become stuck in local optima and be unable to make a short-term sacrifice for a long-term gain. The future of planning would likely involve hybrid approaches that com-

bine the LM's high-level, semantic understanding with more traditional, structured planning algorithms such as Monte Carlo Tree Search [259]. Developing agents that can build and simulate consequences within their own internal world models is crucial for enabling true strategic depth [260]. For instance, on TravelPlanner, even the strongest models achieve only a small fraction of compositional constraint satisfaction, with GPT-4-Turbo reaching merely 4.4% final pass rate when all information is provided upfront, compared to 0.6% in the standard setting [141]. Flex-TravelPlanner further reveals that both GPT-4o and LLaMA 3.1 perform better when global constraints are introduced after local ones [142], suggesting that agents struggle with systematic constraint prioritization. These results expose three recurring failure modes. **(i)** *Sub-plan invalidation cascade*: A single unexpected outcome in a reactive framework invalidates the entire pre-computed plan, requiring complete replanning. **(ii)** *Constraint-order sensitivity*: Agents lack the ability to systematically prioritize constraints, causing success to depend on arbitrary presentation order rather than intrinsic problem structure. **(iii)** *Local optima trapping*: Purely reactive agents reason one step ahead and cannot evaluate trade-offs requiring short-term sacrifice for long-term gain. The root cause is the absence of a persistent world model that supports lookahead evaluation and constraint-aware plan repair. We identify the following open questions for the community:

- How can agents perform systematic constraint prioritization and detect infeasibility *before* plan execution, rather than discovering violations through trial and error?
- What hybrid architectures combine LM semantic planning with classical trajectory optimization to guarantee dynamic feasibility in embodied agents?
- Can plan repair mechanisms be designed that localize failures to sub-plans without requiring full replanning from scratch?

5.3 Open-World Adaptation

The ability of an agent to generalize its skills and adapt to entirely new environments remains a significant hurdle. Many successful agents operate with a pre-defined set of tools or within a known domain. The paradigm of lifelong learning [101] is promising but often slow and computationally intensive. The next frontier is to develop agents capable of zero-shot or few-shot tool use [261], allowing them to learn how to operate a new API or piece of software from its documentation alone. Moreover, addressing key challenges in on-device federated learning such as training instability through advanced optimization techniques [262] is crucial for realizing the vision of agents that can adapt autonomously in open-ended environments [263]. For instance, Voyager's autonomous skill library enables lifelong learning through compositional skill reuse [101], but its effectiveness depends on consistent environment dynamics, and when the environment changes fundamentally, previously acquired skills may not transfer. On continual learning benchmarks, while specialized mechanisms such as self-synthesized rehearsal [113] and progressive prompt optimization [264] demonstrate significant reductions in catastrophic forgetting compared to naive fine-tuning, substantial gaps remain when scaling to longer sequences of diverse tasks in interactive environments. These limitations reveal why current lifelong learning approaches remain slow and brittle. **(i)** *Slow adaptation*: Skill acquisition paradigms require extensive environment interaction and computational resources that are infeasible for real-world deployment. **(ii)** *Catastrophic forgetting*: Agents lose previously acquired capabilities when learning new tasks because parameter updates overwrite rather than augment existing knowledge, as evidenced on continual-learning benchmarks. **(iii)** *Storage degradation*: Memory retrieval accuracy degrades substantially when conversations must be retrieved from external rather than in-context storage, compounding brittleness over extended deployment. We identify the following open questions for the community:

- How can agents learn to operate new tools from documentation alone (zero-shot/few-shot tool use) without task-specific training data?
- What memory architectures prevent catastrophic forgetting while maintaining plasticity for long-deployment agents?
- What cross-modal transfer mechanisms can bridge the sim-to-real gap for embodied VLA models without extensive real-world fine-tuning?

5.4 Safety of LM-Based AI Agents

A critical challenge is ensuring that LM-based AI agents operate safely, ethically, and in alignment with human values [265, 266]. As agents gain more agency in both digital and physical worlds, the potential for unintended consequences or malicious misuse increases. Therefore, ensuring the stability of the underlying neural control systems [267], especially in the presence of inherent time delays in sensing and actuation, is a long-standing and still-critical foundational challenge in this domain [268–270]. This necessitates research into scalable oversight, where a human can effectively supervise multiple agents, and the development of formally verifiable safety constraints to create robust behavioral guardrails. For instance,

End-to-end VLA architectures such as LMDrive [17] collapse perception, reasoning, and control into a single neural function, inherently limiting interpretability and complicating safety auditing. SwarmGPT [251] combines LM-based choreography with distributed safe motion planning for drone swarms, yet LM-generated high-level plans can still introduce errors that cascade into physical settings with irreversible consequences. These cases exemplify three systemic failure modes. **(i)** *Decision opacity*: End-to-end VLA models map sensor observations directly to actions through monolithic transformers, rendering it impossible to audit which input features drove a safety-critical decision. **(ii)** *Dynamical infeasibility*: Semantic decisions from LM planners may not translate into dynamically feasible trajectories, creating a verification gap between high-level intent and low-level execution. **(iii)** *Centralized vulnerability*: Single-point planning architectures lack formally verified emergency overrides, so a single reasoning error can propagate unimpeded to physical action. The root cause is the absence of safety-critical middleware that can hard-intervene when LM outputs violate pre-defined physical constraints. We identify the following open questions for the community:

- How can formally verifiable safety constraints (e.g., control barrier functions) be integrated with LM-based decision-making without excessive conservatism?
- What real-time monitoring mechanisms can detect when an LM-generated plan violates physical safety bounds and trigger guaranteed-safe fallback policies?
- Can interpretable intermediate representations be extracted from end-to-end VLA models for post-hoc safety certification?

5.5 Multi-Agent Collaboration and Social Learning

Beyond the optimization of individual agents, a key frontier lies in the dynamics of multi-agent systems (MASs), which present profound challenges in coordination, negotiation, and conflict resolution [271]. Ensuring that groups of LM-based AI agents can collaborate effectively towards a common goal, especially in decentralized and dynamic environments, remains a significant hurdle [272]. A promising future direction involves endowing agents with social learning capabilities, enabling them to learn not only from direct experience but also by observing, imitating, and communicating with peers [273, 274]. For instance, RoCo's dialectic multi-robot collaboration through natural language dialogue introduces LM-query delays that are not desirable for dynamic or speed-sensitive tasks, and perception inaccuracies can lead to open-loop execution failures [231]. In multi-agent code generation, AgentCoder's fixed three-role architecture (programmer, test-designer, test-executor) demonstrates that coordination overhead increases with task complexity even for small agent teams [165]. These limitations reflect three recurring failure modes. **(i)** *Communication bottleneck*: Multi-turn dialogue for plan coordination introduces latency that scales with agent count, making real-time coordination impossible for time-critical tasks. **(ii)** *Emergent misalignment*: Agents with individually reasonable local objectives drift from the global goal as interaction complexity increases. **(iii)** *Coordination overhead explosion*: The cost of maintaining consistency across agent state grows superlinearly with task complexity and team size. A shared root cause is the absence of lightweight, decentralized consensus protocols that can operate within the inference latency budget of current LMs. We identify the following open questions for the community:

- How can decentralized consensus protocols achieve real-time coordination without a central planner?
- What social learning mechanisms enable agents to acquire skills by observing and imitating peers?
- How can emergent misalignment be detected and corrected before it cascades across the agent group?

5.6 Embodied Deployment Challenges

Deploying LM-based agents in physical environments exposes three interlocking challenges: Fragile perception-control closed loops, violated real-time constraints, and a persistent sim-to-real gap. **Perception-control closed loops** are inherently fragile because the LM's powerful but slow semantic reasoning sits at odds with the physical environment's demand for fast, deterministic control. Perceptual errors propagate through the reasoning module to degrade control reliability. In NaVILA [226], noise in terrain navigation propagates through the reasoning chain to produce incorrect locomotion commands. In VoxPoser [215], perception pipeline errors such as detector sensitivity to object pose and incomplete part segmentation cause manipulation failures. In modular systems such as LM-Nav [61], brittle alignment between perception, reasoning, and control modules allows errors to cascade across the entire pipeline. **Real-time constraints** are systematically violated because visual input tokenization inflates inference latency, exceeding the time budget of real-time control loops. Closed-loop VLA systems re-evaluate control decisions at each timestep through transformer-based architectures, yet the inherent latency of multimodal LM inference creates a mismatch with the physical environment's timing requirements. The **sim-to-real gap** spans multiple

dimensions. Visual domain shift degrades perception at deployment because simulator lighting, textures, and camera parameters differ from reality. Contact dynamics mismatch arises because rigid-body approximations fail to capture deformation, friction variation, and unmodeled compliance. Sample inefficiency makes extensive real-world fine-tuning prohibitively costly. NaVILA [226] shows a fundamental gap when generalizing from simulation to diverse natural terrain. Foundation models such as PaLM-E [74] and RT-2 [75] represent initial steps toward generalizable physical priors, but lightweight adaptation mechanisms remain an open problem. These limitations reflect three recurring failure modes. **(i)** *Perception-to-control latency*: Multimodal LM inference is too slow for real-time control, and visual token inflation compounds the bottleneck. **(ii)** *Cascading noise*: Sensor errors propagate through the reasoning chain to produce incorrect actions. **(iii)** *Domain gap amplification*: Policies trained in simulation overfit to simplified physics and visuals, so deployment to real environments requires extensive re-tuning. We identify the following open questions for the community:

- How can hierarchical architectures separate low-frequency semantic planning from high-frequency reflexive control with guaranteed stability at the interface?
- What token-compression techniques for visual and spatial inputs can reduce multimodal inference latency to meet real-time control requirements without sacrificing grounding accuracy?
- What on-policy adaptation mechanisms can bridge the remaining sim-to-real gap within a small number of physical trials?

6 Conclusions

In this review, we have systematically charted the landscape of LM-based AI agents by deconstructing their modular architecture, rigorously analyzing their evaluation methodologies, and showcasing their transformative applications across both digital and embodied domains. We have identified a decisive shift in the research paradigm: Moving from training specialized models toward designing complex systems that leverage a pre-trained LM as their foundation. This framework has unlocked unprecedented capabilities in problem-solving and automation. However, our analyses have also underscored that significant challenges have persisted. The brittleness of reasoning, the difficulty of long-horizon strategic planning, and the critical need for verifiable safety and alignment have been identified as the primary barriers to reliable, real-world deployment. In conclusion, this survey has provided a foundational overview of the state-of-the-art, highlighting the immense progress that has been made and clarifying the essential works that need further development to build the next generation of AI agents.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant 62476115, in part by Gansu Provincial Department of Education: 2026 Graduate Innovation Star Project “2026CXZX-054”, in part by the Supercomputing Center of Lanzhou University, and in part by the Fundamental Research Funds for the Central Universities under Grant lzujbky-2025-ytB01.

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

[1] Z. Xi *et al.*, The rise and potential of large language model based agents: A survey, *Science China Information Sciences*, vol. 68, no. 2, p. 121101, 2025.

[2] Y. Shi *et al.*, Diffusion models for medical image computing: A survey, *Tsinghua Science and Technology*, vol. 30, no. 1, pp. 357–383, 2025.

[3] G. Cai, R. Tian, L. Yang, Y. Jia, L. Li, and J. Wang, Efficient inference for edge large language models: A survey, *Tsinghua Science and Technology*, vol. 31, no. 3, pp. 1365–1380, 2026.

[4] Z. Durante *et al.*, Agent AI: Surveying the horizons of multimodal interaction, *arXiv preprint arXiv:2401.03568*, 2024.

[5] Y. Yang, X. Wang, R. M. Voyles, and X. Ma, A predefined-time convergent and noise-tolerant zeroing neural network model for time variant quadratic programming with application to robot motion planning, *Tsinghua Science and Technology*, vol. 31, no. 3, pp. 1610–1621, 2026.

[6] Z. Zeng, J. Wang, and X. Liao, Global exponential stability of a general class of recurrent neural networks with time-varying delays, *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 50, no. 10, pp. 1353–1358, 2003.

[7] H. Su, Y. Ye, Y. Qiu, Y. Cao, and M. Z. Q. Chen, Semi-global output consensus for discrete-time switching networked systems subject to input saturation and external disturbances, *IEEE Transactions on Cybernetics*, vol. 49, no. 11, pp. 3934–3945, 2019.

[8] P. Chhikara, D. Khant, S. Aryan, T. Singh, and D. Yadav, Mem0: Building production-ready AI agents with scalable long-term memory, *arXiv preprint arXiv:2504.19413*, 2025.

[9] T. Schick *et al.*, Toolformer: Language models can teach themselves to use tools, in *Advances in Neural Information Processing Systems*, 2023. [Online]. Available: https://openreview.net/forum?id=Yacmpz84TH

[10] A. Madaan *et al.*, Self-Refine: Iterative refinement with self-feedback, in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 46 534–46 594.

[11] L. Wang *et al.*, A survey on large language model based autonomous agents, *Frontiers of Computer Science*, vol. 18, no. 6, p. 186345, 2024.

[12] J. Luo *et al.*, Large language model agent: A survey on methodology, applications and challenges, *arXiv preprint arXiv:2503.21460*, 2025.

[13] M. Cheng *et al.*, A comprehensive survey of the LLM-based agent: The contextual cognition perspective, 2026.

[14] J. Yang *et al.*, SWE-Agent: Agent-computer interfaces enable automated software engineering, in *Advances in Neural Information Processing Systems*, vol. 37, 2024, pp. 50 528–50 652.

[15] T. H. Trinh, Y. Wu, Q. V. Le, H. He, and T. Luong, Solving Olympiad geometry without human demonstrations, *Nature*, vol. 625, no. 7995, pp. 476–482, 2024.

[16] R. Mon-Williams, G. Li, R. Long, W. Du, and C. G. Lucas, Embodied large language models enable robots to complete complex tasks in unpredictable environments, *Nature Machine Intelligence*, vol. 7, no. 4, pp. 592–601, 2025.

[17] H. Shao *et al.*, LMDrive: Closed-loop end-to-end driving with large language models, in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 15 120–15 130.

[18] Z. Zhang *et al.*, A survey on the memory mechanism of large language model-based agents, *ACM Transactions on Information Systems*, vol. 43, no. 6, pp. 1–47, 2025.

[19] L. Wang *et al.*, Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models, in *Annual meeting of the association for computational linguistics*, 2023, pp. 2609–2634.

[20] N. Shinn, F. Cassano, A. Gopinath, K. Narasimhan, and S. Yao, Reflexion: Language agents with verbal reinforcement learning, in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 8634–8652.

[21] Y. He *et al.*, Intelligent decision-making driven by large AI models: Progress, challenges and prospects, *CAAI Transactions on Intelligence Technology*, vol. 10, no. 6, pp. 1573–1592, 2025.

[22] P. Wang, Z. Zhu, Q. Chen, and W. Dai, Text reasoning chain extraction for multi-hop question answering, *Tsinghua Science and Technology*, vol. 29, no. 4, pp. 959–970, 2024.

[23] S. Goyal and S. Dan, IOLBENCH: Benchmarking LLMs on linguistic reasoning, *arXiv preprint arXiv:2501.04249*, 2025.

[24] Y. Zhu *et al.*, ChatNav: Leveraging LLM to zero-shot semantic reasoning in object navigation, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 35, no. 3, pp. 2369–2381, 2025.

[25] A. Toroghi, A. Pesaranghader, T. Sadhu, and S. Sanner, LLM-based typed hyperresolution for commonsense reasoning with knowledge bases, in *International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=wNobG8bV5Q

[26] A. Toroghi, W. Guo, A. Pesaranghader, and S. Sanner, Verifiable, debuggable, and repairable commonsense logical reasoning via LLM-based theory resolution, in *Conference on Empirical Methods in Natural Language Processing*, 2024, pp. 6634–6652.

[27] H. Liu *et al.*, Logical reasoning in large language models: A survey, *arXiv preprint arXiv:2502.09100*, 2025.

[28] J. Ahn, R. Verma, R. Lou, D. Liu, R. Zhang, and W. Yin, Large language models for mathematical reasoning: Progresses and challenges, in *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, 2024, pp. 225–237. [Online]. Available: https://aclanthology.org/2024.eacl-srw.17/

[29] E. Kiciman, R. Ness, A. Sharma, and C. Tan, Causal reasoning and large language models: Opening a new frontier for causality, *Transactions on Machine Learning Research*, 2024. [Online]. Available: https://openreview.net/forum?id=mqoxLkX210

[30] H. Chi *et al.*, Unveiling causal reasoning in large language models: Reality or mirage? in *Advances in Neural Information Processing Systems*, 2024. [Online]. Available: https://openreview.net/forum?id=1IU3P8VDbn

[31] K. Gandhi, J.-P. Fraenken, T. Gerstenberg, and N. Goodman, Understanding social reasoning in language models with language models, in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 13 518–13 529.

[32] J. Lee, Y. Wang, J. Li, and M. Zhang, Multimodal reasoning with multimodal knowledge graph, in *Meeting of the Association for Computational Linguistics*, Bangkok, Thailand, 2024, pp. 10 767–10 782.

[33] U. Agarwal, K. Tanmay, A. Khandelwal, and M. Choudhury, Ethical reasoning and moral value alignment of LLMs depend on the language we prompt them in, in *Conference on Computational Linguistics, Language Resources and Evaluation*, 2024, pp. 6330–6340.

[34] Z. Ke *et al.*, A survey of frontiers in LLM reasoning: Inference scaling, learning to reason, and agentic systems, *Transactions on Machine Learning Research*, 2025. [Online]. Available: https://openreview.net/forum?id=SlsZZ25InC

[35] J. Wei *et al.*, Chain of thought prompting elicits reasoning in large language models, in *Advances in Neural Information Processing Systems*, 2022. [Online]. Available: https://openreview.net/forum?id=_VjQlMeSB_J

[36] X. Wang *et al.*, Self-consistency improves chain of thought reasoning in language models, in *International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=1PL1NIMMrw

[37] S. Yao *et al.*, Tree of thoughts: Deliberate problem solving with large language models, in *Advances in Neural*

Information Processing Systems, 2023. [Online]. Available: https://openreview.net/forum?id=5Xc1ecxO1h

[38] K. Cobbe *et al.*, Training verifiers to solve math word problems, *arXiv preprint arXiv:2110.14168*, 2021.

[39] L. Yu *et al.*, MetaMath: Bootstrap your own mathematical questions for large language models, in *International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=N8N0hgNDRt

[40] D. Silver *et al.*, Mastering the game of Go with deep neural networks and tree search, *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[41] D. Guo *et al.*, DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning, *Nature*, vol. 645, no. 8081, pp. 633–638, 2025.

[42] H. Lightman *et al.*, Let's verify step by step, in *International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=v8L0pN6EOi

[43] L. Ouyang *et al.*, Training language models to follow instructions with human feedback, in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 27 730–27 744.

[44] A. Vaswani *et al.*, Attention is all you need, in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[45] Z. Zong *et al.*, Accelerating distributed training of large concurrent-branch models through bidirectional pipeline coordination, *Tsinghua Science and Technology*, vol. 30, no. 6, pp. 2638–2652, 2025.

[46] A. Gu and T. Dao, Linear-time sequence modeling with selective state spaces, in *Conference on Language Modeling*, 2024. [Online]. Available: https://openreview.net/forum?id=tEYskw1VY2

[47] B. Lenz *et al.*, Jamba: Hybrid Transformer-Mamba language models, in *International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=JFPaD7lpBD

[48] J. Zhang, C. Xu, S. Shen, J. Zhu, and P. Zhang, MFF-YOLO: An improved YOLO algorithm based on multi-scale semantic feature fusion, *Tsinghua Science and Technology*, vol. 30, no. 5, pp. 2097–2113, 2025.

[49] J. Lu, C. Wu, L. Wang, R. Li, and X. Shen, Dual-modality integration attention with graph-based feature extraction for visual question and answering, *Tsinghua Science and Technology*, vol. 30, no. 5, pp. 2133–2145, 2025.

[50] Z. Xian, R. Huang, D. Towey, and C. Yue, Convolutional neural network image classification based on different color spaces, *Tsinghua Science and Technology*, vol. 30, no. 1, pp. 402–417, 2025.

[51] M. Li *et al.*, LLM-enabled agentic DRL for generic-specialized visual GenAI collaboration in edge–cloud computing networks, *Tsinghua Science and Technology*, 2026.

[52] A. Dosovitskiy *et al.*, An image is worth 16x16 words: Transformers for image recognition at scale, in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy

[53] A. Radford *et al.*, Learning transferable visual models from natural language supervision, in *International Conference on Machine Learning*, vol. 139, 2021, pp. 8748–8763.

[54] H. Liu, C. Li, Q. Wu, and Y. J. Lee, Visual instruction tuning, in *Advances in Neural Information Processing Systems*, 2023. [Online]. Available: https://openreview.net/forum?id=w0H2xGHlkw

[55] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. Mcleavey, and I. Sutskever, Robust speech recognition via large-scale weak supervision, in *International Conference on Machine Learning*, vol. 202, 2023, pp. 28 492–28 518.

[56] H. Dinkel *et al.*, MiDashengLM: Efficient audio understanding with general audio captions, *arXiv preprint arXiv:2508.03983*, 2025.

[57] C. Wang *et al.*, Opens2s: Advancing fully open-source end-to-end empathetic large speech language model, in *Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2025, pp. 906–917.

[58] Z. Long *et al.*, VITA-Audio: Fast interleaved cross-modal token generation for efficient large speech-language model, *arXiv preprint arXiv:2505.03739*, 2025.

[59] D. Liang *et al.*, Evolution of laser technology for automotive LiDAR, an industrial viewpoint, *Nature Communications*, vol. 15, no. 1, p. 7660, 2024.

[60] R. Horaud, M. Hansard, G. Evangelidis, and C. Ménier, An overview of depth cameras and range scanners based on time-of-flight technologies, *Machine Vision and Applications*, vol. 27, no. 7, pp. 1005–1020, 2016.

[61] D. Shah, B. Osiński, S. Levine *et al.*, LM-Nav: Robotic navigation with large pre-trained models of language, vision, and action, in *Conference on Robot Learning*. PMLR, 2023, pp. 492–504.

[62] J. Luo *et al.*, DSPNet: Dual-vision scene perception for robust 3D question answering, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 14 169–14 178.

[63] J. Hwangbo *et al.*, Learning agile and dynamic motor skills for legged robots, *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[64] L. Seminara *et al.*, Active haptic perception in robots: A review, *Frontiers in Neurorobotics*, vol. 13, p. 53, 2019.

[65] G. Hu, D. Hershcovich, and H. Seifi, HapticLLaMA: A multimodal sensory language model for haptic captioning, in *Findings of the Association for Computational Linguistics: EACL 2026*, 2026, pp. 3180–3192.

[66] L. F. M. Fuentes *et al.*, VLH: Vision-language-haptics foundation model, *arXiv preprint arXiv:2508.01361*, 2025.

[67] F. R. Hogan, J. Ballester, S. Dong, and A. Rodriguez, Tactile dexterity: Manipulation primitives with tactile feedback, in *International Conference on Robotics and Automation*, 2020, pp. 8863–8869.

[68] Y. Li, X. Wei, Y. Zhou, J. Wang, and R. You, Research progress of electronic nose technology in exhaled breath

disease analysis, *Microsystems & Nanoengineering*, vol. 9, no. 1, p. 129, 2023.

[69] P. Sorino, G. M. Biancofiore, D. Lofù, T. Colafiglio, A. Lombardi, F. Narducci, and T. Di Noia, ARIEL: Brain-computer interfaces meet large language models for emotional support conversation, in *ACM Conference on User Modeling, Adaptation and Personalization*, 2024, pp. 601–609.

[70] R. Zhang *et al.*, NOIR: Neural signal operated intelligent robots for everyday activities, in *Conference on Robot Learning*, 2023. [Online]. Available: https://openreview.net/forum?id=eyykI3UIHa

[71] R. Xiong, S. Zhao, C. Chen, and Z. Xu, Optimizing multimodal data queries in data lakes, *Tsinghua Science and Technology*, vol. 30, no. 6, pp. 2625–2637, 2025.

[72] G. Huang, D. H. Tsang, S. Yang, G. Lei, and L. Liu, Cued-Agent: A collaborative multi-agent system for automatic cued speech recognition, in *ACM International Conference on Multimedia*, 2025, pp. 8313–8321.

[73] W. Li, J. Zhao, L. Zhu, L. Zhang, and H. Wang, TopoPharmDTI: Improving interactions prediction by enhanced deep learning representation for both drug and target molecules, *Tsinghua Science and Technology*, vol. 31, no. 1, pp. 399–417, 2024.

[74] D. Driess *et al.*, PaLM-E: An embodied multimodal language model, in *International Conference on Machine Learning*, vol. 202, 2023, pp. 8469–8488.

[75] B. Zitkovich *et al.*, RT-2: Vision-language-action models transfer web knowledge to robotic control, in *Conference on Robot Learning*, vol. 229, 2023, pp. 2165–2183.

[76] K. Mei *et al.*, AIOS: LLM agent operating system, in *Conference on Language Modeling*, 2025. [Online]. Available: https://openreview.net/forum?id=L4HHkCDz2x

[77] B. Wang *et al.*, SCM: Enhancing large language model with self-controlled memory framework, in *International Conference on Database Systems for Advanced Applications*, 2025, pp. 188–203.

[78] W. Zhong, L. Guo, Q. Gao, H. Ye, and Y. Wang, MemoryBank: Enhancing large language models with long-term memory, in *AAAI Conference on Artificial Intelligence*, vol. 38, no. 17, 2024, pp. 19 724–19 731.

[79] M. Hu, T. Chen, Q. Chen, Y. Mu, W. Shao, and P. Luo, HiAgent: Hierarchical working memory management for solving long-horizon agent tasks with large language model, in *Annual Meeting of the Association for Computational Linguistics*, 2025, pp. 32 779–32 798.

[80] N. Liu *et al.*, From LLM to conversational agent: A memory enhanced architecture with fine-tuning of large language models, *arXiv preprint arXiv:2401.02777*, 2024.

[81] Z. Liu *et al.*, AgentLite: A lightweight library for building and advancing task-oriented LLM agent system, *arXiv preprint arXiv:2402.15538*, 2024.

[82] G. Sarch, Y. Wu, M. Tarr, and K. Fragkiadaki, Open-ended instructable embodied agents with memory-augmented large language models, in *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023, pp. 3468–3500.

[83] Y. Shang *et al.*, AgentSquare: Automatic LLM agent search in modular design space, in *International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=mPdmDYIQ7f

[84] Z. Wang *et al.*, JARVIS-1: Open-world multi-task agents with memory-augmented multimodal language models, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 47, no. 3, pp. 1894–1907, 2025.

[85] H. Sun, S. Zeng, and B. Zhang, H-MEM: Hierarchical memory for high-efficiency long-term reasoning in LLM agents, in *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2026, pp. 341–350.

[86] J. Zhu, J. Li, C. Zhang, J. Liu, and M. Yang, HeLa-Mem: Hebbian learning and associative memory for LLM agents, *arXiv preprint arXiv:2604.16839*, 2026.

[87] J. S. Park, J. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein, Generative agents: Interactive simulacra of human behavior, in *ACM Symposium on User Interface Software and Technology*, 2023.

[88] S. Yao *et al.*, ReAct: Synergizing reasoning and acting in language models, in *International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=WE_vluYUL-X

[89] A. Diko, T. Wang, W. Swaileh, S. Sun, and I. Patras, ReWind: Understanding long videos with instructed learnable memory, in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 13 734–13 743.

[90] W. Liu, Z. Tang, J. Li, K. Chen, and M. Zhang, MemLong: Memory-augmented retrieval for long text modeling, *arXiv preprint arXiv:2408.16967*, 2024.

[91] E. Song *et al.*, MovieChat: From dense token to sparse memory for long video understanding, in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 18 221–18 232.

[92] H. Zhang *et al.*, Flash-VStream: Memory-based real-time understanding for long video streams, *arXiv preprint arXiv:2406.08085*, 2024.

[93] L. Long *et al.*, Seeing, listening, remembering, and reasoning: A multimodal agent with long-term memory, *arXiv preprint arXiv:2508.09736*, 2025.

[94] F. Ocker, J. Deigmöller, P. Smirnov, and J. Eggert, A grounded memory system for smart personal assistants, *arXiv preprint arXiv:2505.06328*, 2025.

[95] W. Xu, Z. Liang, K. Mei, H. Gao, J. Tan, and Y. Zhang, A-Mem: Agentic memory for LLM agents, in *Advances in Neural Information Processing Systems*, vol. 38, 2025, pp. 17 577–17 604.

[96] J. Kang, M. Ji, Z. Zhao, and T. Bai, Memory OS of AI agent, in *Conference on Empirical Methods in Natural Language Processing*, 2025, pp. 25 972–25 981.

[97] W. Fan *et al.*, A survey on RAG meeting LLMs: Towards retrieval-augmented large language models, in *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 6491–6501.

[98] Z. Jiang *et al.*, Active retrieval augmented generation, in *Conference on Empirical Methods in Natural Language Processing*, 2023, pp. 7969–7992.

[99] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, Self-RAG: Learning to retrieve, generate, and critique through self-reflection, in *International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=hSyW5go0v8

[100] Y. Shen *et al.*, HuggingGPT: Solving AI tasks with ChatGPT and its friends in Hugging Face, in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 38 154–38 180.

[101] G. Wang *et al.*, Voyager: An open-ended embodied agent with large language models, *Transactions on Machine Learning Research*, 2024. [Online]. Available: https://openreview.net/forum?id=ehfRiF0R3a

[102] J. Liang *et al.*, Code as policies: Language model programs for embodied control, in *IEEE International Conference on Robotics and Automation*, 2023, pp. 9493–9500.

[103] K. Saab *et al.*, Capabilities of Gemini models in medicine, *arXiv preprint arXiv:2404.18416*, 2024.

[104] Q. Zhu *et al.*, Deepseek-Coder-V2: Breaking the barrier of closed-source models in code intelligence, *arXiv preprint arXiv:2406.11931*, 2024.

[105] X. Deng *et al.*, Mind2Web: Towards a generalist agent for the web, in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 28 091–28 114.

[106] H. Luo *et al.*, WizardMath: Empowering mathematical reasoning for large language models via reinforced Evol-Instruct, in *International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=mMPMHWOdOy

[107] L. Wang, X. Zhang, H. Su, and J. Zhu, A comprehensive survey of continual learning: Theory, method and application, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 8, pp. 5362–5383, 2024.

[108] E. J. Hu *et al.*, LoRA: Low-rank adaptation of large language models, in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=nZeVKeeFYf9

[109] Z. Wang *et al.*, Rehearsal-free continual language learning via efficient parameter isolation, in *Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023, pp. 10 933–10 946.

[110] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, Direct preference optimization: Your language model is secretly a reward model, in *Advances in Neural Information Processing Systems*, 2023. [Online]. Available: https://openreview.net/forum?id=HPuSIXJaa9

[111] M. Ye, W. Huang, Z. Shi, Z. Ye, and B. Du, Biamix contrastive learning and memory similarity distillation in class-incremental learning, *CAAI Transactions on Intelligence Technology*, vol. 10, no. 6, pp. 1745–1758, 2025.

[112] X. Wang *et al.*, Orthogonal subspace learning for language model continual learning, in *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023, pp. 10 658–10 671.

[113] J. Huang *et al.*, Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal, in *Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024, pp. 1416–1428.

[114] J. Zheng, S. Qiu, C. Shi, and Q. Ma, Towards lifelong learning of large language models: A survey, *ACM Comput. Surv.*, vol. 57, no. 8, 2025.

[115] S. Guo *et al.*, Direct language model alignment from online AI feedback, *arXiv preprint arXiv:2402.04792*, 2024.

[116] H. Dong *et al.*, RLHF workflow: From reward modeling to online RLHF a comprehensive practical alignment recipe of iterative preference learning, *Transactions on Machine Learning Research*, vol. 2024, 2024.

[117] S. Zhou *et al.*, WebArena: A realistic web environment for building autonomous agents, in *International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=oKn9c6ytLx

[118] J. Y. Koh *et al.*, VisualWebArena: Evaluating multimodal agents on realistic visual web tasks, in *Meeting of the Association for Computational Linguistics*, 2024, pp. 881–905.

[119] J. Wei *et al.*, BrowseComp: A simple yet challenging benchmark for browsing agents, *arXiv preprint arXiv:2504.12516*, 2025.

[120] T. Xue *et al.*, An illusion of progress? assessing the current state of web agents, in *Conference on Language Modeling*, 2025. [Online]. Available: https://openreview.net/forum?id=6jZi4HSs6o

[121] A. Gu, B. Roziere, H. J. Leather, A. Solar-Lezama, G. Synnaeve, and S. Wang, CRUXEval: A benchmark for code reasoning, understanding and execution, in *International Conference on Machine Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=Ffpg52swvg

[122] C. E. Jimenez, J. Yang, A. Wettig, S. Yao, K. Pei, O. Press, and K. R. Narasimhan, SWE-bench: Can language models resolve real-world GitHub issues? in *International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=VTF8yNQM66

[123] T. Y. Zhuo *et al.*, BigCodeBench: Benchmarking code generation with diverse function calls and complex instructions, in *International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=YrycTjllL0

[124] N. Edwards, Y. Lee, Y. A. Mao, Y. Qin, S. Schuster, and N. Kim, RExBench: Can coding agents au-

tonomously implement AI research extensions? *arXiv preprint arXiv:2506.22598*, 2025.

[125] T. Xie *et al.*, OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments, in *Advances in Neural Information Processing Systems*, vol. 37, 2024, pp. 52 040–52 094.

[126] T. Kuntz *et al.*, OS-Harm: A benchmark for measuring safety of computer use agents, in *Advances in Neural Information Processing Systems*, vol. 38, 2025.

[127] R. Bonatti *et al.*, Windows Agent Arena: Evaluating multi-modal OS agents at scale, in *International Conference on Machine Learning*, vol. 267, 2025, pp. 4874–4910.

[128] C. Rawles, A. Li, D. Rodriguez, O. Riva, and T. Lillicrap, AndroidInTheWild: A large-scale dataset for Android device control, in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 59 708–59 728.

[129] Y. Qin *et al.*, ToolLLM: Facilitating large language models to master 16000+ real-world APIs, in *International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=dHng2O0Jjr

[130] K. Basu *et al.*, NESTFUL: A benchmark for evaluating LLMs on nested sequences of API calls, in *Conference on Empirical Methods in Natural Language Processing*, 2025, pp. 33 538–33 547.

[131] S. Yao, N. Shinn, P. Razavi, and K. R. Narasimhan, τ-bench: A benchmark for tool-agent-user interaction in real-world domains, in *International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=roNSXZpUDN

[132] X. Liu *et al.*, AgentBench: Evaluating LLMs as agents, in *International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=zAdUB0aCTQ

[133] Z. Xi *et al.*, AgentGym: Evaluating and training large language model-based agents across diverse environments, in *Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 27 914–27 961.

[134] J. Zheng *et al.*, LifelongAgentBench: Evaluating LLM agents as lifelong learners, *arXiv preprint arXiv:2505.11942*, 2025.

[135] O. Hofman *et al.*, MAPS: A multilingual benchmark for agent performance and security, in *Findings of the Association for Computational Linguistics: EACL 2026*, Mar. 2026, pp. 821–845.

[136] Z. Duan *et al.*, SkillAttack: Automated red teaming of agent skills through attack path refinement, *arXiv preprint arXiv:2604.04989*, 2026.

[137] E. Debenedetti, J. Zhang, M. Balunovic, L. Beurer-Kellner, M. Fischer, and F. Tramèr, AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents, in *Advances in Neural Information Processing Systems*, vol. 37, 2024, pp. 82 895–82 920.

[138] Z. Zhang *et al.*, Agent-SafetyBench: Evaluating the safety of LLM agents, *arXiv preprint arXiv:2412.14470*, 2024.

[139] Y. Yang, Y. Wang, D. Li, Z. Luo, B. Chen, C. Huang, and J. Li, Aria-UI: Visual grounding for GUI instructions, in *Findings of the Association for Computational Linguistics: ACL 2025*, 2025, pp. 22 418–22 433.

[140] Z. Li *et al.*, MemOS: A memory OS for AI system, *arXiv preprint arXiv:2507.03724*, 2025.

[141] J. Xie *et al.*, TravelPlanner: A benchmark for real-world planning with language agents, in *International Conference on Machine Learning*, vol. 235, 2024, pp. 54 590–54 613.

[142] J. Oh, E. Kim, and A. Oh, Flex-TravelPlanner: A benchmark for flexible planning with language agents, in *Workshop on Reasoning and Planning for Large Language Models*, 2025. [Online]. Available: https://openreview.net/forum?id=a7unQ5jMx7

[143] F. Lei, Y. Yang, W. Sun, and D. Lin, MCPVerse: An expansive, real-world benchmark for agentic tool use, *arXiv preprint arXiv:2508.16260*, 2025.

[144] I. Gur *et al.*, A real-world WebAgent with planning, long context understanding, and program synthesis, in *International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=9JQtrumvg8

[145] Z. Qi *et al.*, WebRL: Training LLM web agents via self-evolving online curriculum reinforcement learning, in *International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=oVKEAFjEqv

[146] X. H. Lu, Z. Kasner, and S. Reddy, WebLINX: Real-world website navigation with multi-turn dialogue, in *International Conference on Machine Learning*, vol. 235, 2024, pp. 33 007–33 056.

[147] H. Lai *et al.*, AutoWebGLM: A large language model-based web navigating agent, in *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 5295–5306.

[148] K. Cheng *et al.*, SeeClick: Harnessing GUI grounding for advanced visual GUI agents, in *Meeting of the Association for Computational Linguistics*, 2024, pp. 9313–9332.

[149] H. He *et al.*, WebVoyager: Building an end-to-end web agent with large multimodal models, in *Meeting of the Association for Computational Linguistics*, 2024, pp. 6864–6890.

[150] C. Zhang *et al.*, AppAgent: Multimodal agents as smartphone users, in *CHI Conference on Human Factors in Computing Systems*, 2025.

[151] J. Wang *et al.*, Mobile-Agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration, in *Advances in Neural Information Processing Systems*, vol. 37, 2024, pp. 2686–2710.

[152] K. You *et al.*, Ferret-UI: Grounded mobile UI understanding with multimodal LLMs, in *European Conference on Computer Vision*, 2024, pp. 240–255.

[153] Z. LI *et al.*, Ferret-UI 2: Mastering universal user interface understanding across platforms, in *International Conference*

on Learning Representations, 2025. [Online]. Available: https://openreview.net/forum?id=GBfYgjOfSe

[154] Y. Xu *et al.*, Aguvis: Unified pure vision agents for autonomous GUI interaction, in *International Conference on Machine Learning*, 2025. [Online]. Available: https://openreview.net/forum?id=PlihOwfx4r

[155] K. Q. Lin *et al.*, ShowUI: One vision-language-action model for gui visual agent, in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 19 498–19 508.

[156] K. Ma, H. Zhang, H. Wang, X. Pan, and D. Yu, LASER: LLM agent with state-space exploration for web navigation, in *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023. [Online]. Available: https://openreview.net/forum?id=sYFFyAILy7

[157] X. Hu *et al.*, Os Agents: A survey on MLLM-based agents for computer, phone and browser use, in *Meeting of the Association for Computational Linguistics*, 2025, pp. 7436–7465.

[158] Y. Yang *et al.*, Agentic Web: Weaving the next web with AI agents, *arXiv preprint arXiv:2507.21206*, 2025.

[159] I. L. Iong *et al.*, OpenWebAgent: An open toolkit to enable web agents on large language models, in *Meeting of the Association for Computational Linguistics*, 2024, pp. 72–81.

[160] B. Roziere *et al.*, Code Llama: Open foundation models for code, *arXiv preprint arXiv:2308.12950*, 2023.

[161] B. Hui *et al.*, Qwen2.5-Coder technical report, *arXiv preprint arXiv:2409.12186*, 2024.

[162] K. Zhang, J. Li, G. Li, X. Shi, and Z. Jin, CodeAgent: Enhancing code generation with tool-integrated agent systems for real-world repo-level coding challenges, in *Meeting of the Association for Computational Linguistics*, 2024, pp. 13 643–13 658.

[163] T. Zheng *et al.*, OpenCodeInterpreter: Integrating code generation with execution and refinement, in *Association for Computational Linguistics ACL*, 2024, pp. 12 834–12 859.

[164] M. A. Islam, M. E. Ali, and M. R. Parvez, MapCoder: Multi-agent code generation for competitive problem solving, in *Annual Meeting of the Association of Computational Linguistics 2024*. Association for Computational Linguistics, 2024, pp. 4912–4944.

[165] D. Huang, J. M. Zhang, M. Luck, Q. Bu, Y. Qing, and H. Cui, AgentCoder: Multi-agent-based code generation with iterative testing and optimisation, *arXiv preprint arXiv:2312.13010*, 2023.

[166] H. Xin *et al.*, DeepSeek-Prover: Advancing theorem proving in LLMs through large-scale synthetic data, *arXiv preprint arXiv:2405.14333*, 2024.

[167] K. Baba, C. Liu, S. Kurita, and A. Sannai, Prover Agent: An agent-based framework for formal mathematical proofs, in *2nd AI for Math Workshop @ ICML 2025*, 2025. [Online]. Available: https://openreview.net/forum?id=sPdQfGQccH

[168] Z. Gou *et al.*, ToRA: A tool-integrated reasoning agent for mathematical problem solving, in *International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=Ep0TtjVoap

[169] Z. Azerbayev *et al.*, Llemma: An open language model for mathematics, in *International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=4WnqRR915j

[170] Z. Shao *et al.*, DeepSeekMath: Pushing the limits of mathematical reasoning in open language models, *arXiv preprint arXiv:2402.03300*, 2024.

[171] A. Yang *et al.*, Qwen2.5-Math technical report: Toward mathematical expert model via self-improvement, *arXiv preprint arXiv:2409.12122*, 2024.

[172] S. Toshniwal, W. Du, I. Moshkov, B. Kisacanin, A. Ayrapetyan, and I. Gitman, OpenMathInstruct-2: Accelerating AI for math with massive open-source instruction data, in *International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=mTCbq2QssD

[173] P. Wang *et al.*, Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations, in *Association for Computational Linguistics*, 2024, pp. 9426–9439.

[174] K. Zhang *et al.*, UltraMedical: Building specialized generalists in biomedicine, in *Advances in Neural Information Processing Systems*, vol. 37, 2024, pp. 26 045–26 081.

[175] V. Nath *et al.*, VILA-M3: Enhancing vision-language models with medical expert knowledge, in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 14 788–14 798.

[176] J. Wu *et al.*, Medical Graph RAG: Evidence-based medical large language model via graph retrieval-augmented generation, in *Association for Computational Linguistics*, 2025, pp. 28 443–28 467.

[177] N. Khandekar *et al.*, MedCalc-Bench: Evaluating large language models for medical calculations, in *Advances in Neural Information Processing Systems*, vol. 37, 2024, pp. 84 730–84 745.

[178] B. Wang *et al.*, LLM4DEU: Fine tuning large language model for medical diagnosis in outpatient and emergency department visits of neurosurgery, *Tsinghua Science and Technology*, vol. 30, no. 6, pp. 2487–2504, 2025.

[179] J. Chen *et al.*, A privacy policy text compliance reasoning framework with large language models for healthcare services, *Tsinghua Science and Technology*, vol. 30, no. 4, pp. 1831–1845, 2025.

[180] B. Gao *et al.*, Leveraging large language models to enhance medical text representation for lung diagnosis prediction via knowledge infusion, *Tsinghua Science and Technology*, vol. 31, no. 1, pp. 418–429, 2026.

[181] Y. Kim *et al.*, MDAgents: An adaptive collaboration of LLMs for medical decision-making, in *Advances in Neural Information Processing Systems*, vol. 37, 2024, pp. 79 410–79 452.

[182] Z. Wang, A. Li, Z. Li, and X. Liu, GenArtist: Multimodal

LLM as an agent for unified image generation and editing, in *Advances in Neural Information Processing Systems*, vol. 37, 2024, pp. 128 374–128 395.

[183] C. Jia *et al.*, ChatGen: Automatic text-to-image generation from freestyle chatting, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 13 284–13 293.

[184] X. Xu *et al.*, MM-StoryAgent: Immersive narrated storybook video generation with a multi-agent paradigm across text, image and audio, *arXiv preprint arXiv:2503.05242*, 2025.

[185] Z. Liu *et al.*, MagicQuill: An intelligent interactive image editing system, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 13 072–13 082.

[186] Q. Zhang *et al.*, Towards text-guided 3D scene composition, in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 6829–6838.

[187] S. Bahmani *et al.*, TC4D: Trajectory-conditioned text-to-4D generation, in *European Conference on Computer Vision*. Springer, 2024, pp. 53–72.

[188] Z. Yue, S. Zhuang, K. Li, Y. Ding, and Y. Wang, V-Stylist: Video stylization via collaboration and reflection of MLLM agents, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 3195–3205.

[189] J. Wu *et al.*, DiffSensei: Bridging multi-modal LLMs and diffusion models for customized manga generation, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 28 684–28 693.

[190] D. Yu *et al.*, MusicAgent: An AI agent for music understanding and generation with large language models, in *Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2023, pp. 246–255.

[191] E. Karystinaios, WeaveMuse: An open agentic system for multimodal music understanding and generation, in *1st Workshop on Large Language Models for Music & Audio (LLM4MA)*, 2025. [Online]. Available: https://openreview.net/forum?id=03a2HXqUco

[192] Y. Zhang, A. Maezawa, G. Xia, K. Yamamoto, and S. Dixon, Loop Copilot: Conducting AI ensembles for music generation and iterative editing, *arXiv preprint arXiv:2310.12404*, 2023.

[193] Z. Xie, Q. He, Y. Zhu, Q. He, and M. Li, FilmComposer: LLM-driven music production for silent film clips, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 13 519–13 528.

[194] Y. Rong, J. Wang, G. Lei, S. Yang, and L. Liu, AudioGenie: A training-free multi-agent framework for diverse multimodality-to-multiaudio generation, in *ACM International Conference on Multimedia*, 2025, pp. 8872–8881.

[195] X. Zhu *et al.*, Ghost in the Minecraft: Generally capable agents for open-world environments via large language models with text-based knowledge and memory, *arXiv preprint arXiv:2305.17144*, 2023.

[196] S. Liu *et al.*, ODYSSEY: empowering Minecraft agents with open-world skills, in *International Joint Conference on Artificial Intelligence*, 2025.

[197] H. Li *et al.*, Auto MC-Reward: Automated dense reward design with large language models for Minecraft, in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 16 426–16 435.

[198] Y. Qin *et al.*, MP5: A multi-modal open-ended embodied system in Minecraft via active perception, in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 16 307–16 316.

[199] R. Gong *et al.*, MindAgent: Emergent gaming interaction, in *Association for Computational Linguistics: NAACL*, 2024, pp. 3154–3183.

[200] A. Bakhtin *et al.*, Human-level play in the game of Diplomacy by combining language models with strategic reasoning, *Science*, vol. 378, no. 6624, pp. 1067–1074, 2022.

[201] Z. Guan, X. Kong, F. Zhong, and Y. Wang, Richelieu: Self-evolving LLM-based agents for AI diplomacy, in *Advances in Neural Information Processing Systems*, vol. 37, 2024, pp. 123 471–123 497.

[202] X. Shao, W. Jiang, F. Zuo, and M. Liu, SwarmBrain: Embodied agent for real-time strategy game StarCraft II via large language models, *arXiv preprint arXiv:2401.17749*, 2024.

[203] S. Oh, I. Chung, and K.-J. Kim, LangBirds: An agent for Angry Birds using a large language model, in *IEEE Conference on Games*, 2024, pp. 1–8.

[204] M. U. Nasir, S. James, and J. Togelius, Word2World: Generating stories and worlds through large language models, *arXiv preprint arXiv:2405.06686*, 2024.

[205] M. Kim, Y. Jung, D. Lee, and S.-w. Hwang, PLM-based world models for text-based games, in *Conference on Empirical Methods in Natural Language Processing*, 2022, pp. 1324–1341.

[206] D. A. Boiko, R. MacKnight, B. Kline, and G. Gomes, Autonomous chemical research with large language models, *Nature*, vol. 624, no. 7992, pp. 570–578, 2023.

[207] A. Novikov *et al.*, AlphaEvolve: A coding agent for scientific and algorithmic discovery, *arXiv preprint arXiv:2506.13131*, 2025.

[208] A. M. Bran *et al.*, Augmenting large language models with chemistry tools, *Nature machine intelligence*, vol. 6, no. 5, pp. 525–535, 2024.

[209] B. Zhang, X. Li, H. Xu, Z. Jin, Q. Wu, and C. Li, TopoMAS: Large language model driven topological materials multi-agent system, *Materials Genome Engineering Advances*, vol. 4, no. 1, p. e70045, 2026.

[210] C. Zhang and B. I. Yakobson, MatClaw: An autonomous code-first LLM agent for end-to-end materials exploration, *arXiv preprint arXiv:2604.02688*, 2026.

[211] Y. Liu *et al.*, EIoT: Embodied intelligence of things, *Tsinghua Science and Technology*, vol. 31, no. 1, pp. 1–15, 2026.

[212] S. Wang, M. N. Nikolić, T. L. Lam, Q. Gao, R. Ding, and T. Zhang, Robot manipulation based on embodied visual perception: A survey, *CAAI Transactions on Intelligence Technology*, vol. 10, no. 4, pp. 945–958, 2025.

[213] T. Wang, C. Chen, C. Liu, Z. Wang, L. Cheng, and H. Wan, Large language model enhanced intelligent robotic control software development, *Tsinghua Science and Technology*, 2026.

[214] Z. Pang, Y. Hu, J. Yu, B. Zhang, and L. Gong, Discrete data-driven position and orientation control for redundant manipulators with Jacobian matrix learning, *Tsinghua Science and Technology*, vol. 30, no. 5, pp. 1980–1993, 2025.

[215] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei, VoxPoser: Composable 3D value maps for robotic manipulation with language models, in *Proceedings of The 7th Conference on Robot Learning*, vol. 229, 2023, pp. 540–562.

[216] W. Zhou *et al.*, PhysVLM: Enabling visual language models to understand robotic physical reachability, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 6940–6949.

[217] I. Singh *et al.*, ProgPrompt: Generating situated robot task plans using large language models, in *IEEE International Conference on Robotics and Automation*, 2023, pp. 11 523–11 530.

[218] C. H. Song *et al.*, LLM-Planner: Few-shot grounded planning for embodied agents with large language models, in *IEEE/CVF International Conference on Computer Vision*, 2023, pp. 2998–3009.

[219] Y. Wu *et al.*, SELP: Generating safe and efficient task plans for robot agents with large language models, in *IEEE International Conference on Robotics and Automation*, 2025, pp. 2599–2605.

[220] Z. Liu, A. Bahety, and S. Song, REFLECT: Summarizing robot experiences for failure explanation and correction, in *Conference on Robot Learning*, 2023. [Online]. Available: https://openreview.net/forum?id=8yTS_nAILxt

[221] Y. J. Ma *et al.*, Eureka: Human-level reward design via coding large language models, in *International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=IEduRUO55F

[222] A. Brohan *et al.*, RT-1: Robotics Transformer for real-world control at scale, *arXiv preprint arXiv:2212.06817*, 2022.

[223] K. Black *et al.*, π_0: A vision-language-action flow model for general robot control, *arXiv preprint arXiv:2410.24164*, 2024.

[224] J. Wu *et al.*, TidyBot: Personalized robot assistance with large language models, *Autonomous Robots*, vol. 47, no. 8, pp. 1087–1102, 2023.

[225] P. Liu *et al.*, OK-robot: What really matters in integrating open-knowledge models for robotics, in *Workshop on Vision-Language Models for Navigation and Manipulation at ICRA*, 2024. [Online]. Available: https://openreview.net/forum?id=269Pr3Uxtm

[226] A.-C. Cheng *et al.*, NaVILA: Legged robot vision-language-action model for navigation, *arXiv preprint arXiv:2412.04453*, 2024.

[227] Z. Ravichandran, V. Murali, M. Tzes, G. J. Pappas, and V. Kumar, SPINE: Online semantic planning for missions with incomplete natural language specifications in unstructured environments, in *IEEE International Conference on Robotics and Automation*, 2025, pp. 13 714–13 721.

[228] A. J. Sathyamoorthy *et al.*, CoNVOI: Context-aware navigation using vision language models in outdoor and indoor environments, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2024, pp. 13 837–13 844.

[229] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher, VLFM: Vision-language frontier maps for zero-shot semantic navigation, in *IEEE International Conference on Robotics and Automation*, 2024, pp. 42–48.

[230] J. Liu, Z. Jiang, X. Xu, W. Li, M. Liu, and H. Liu, Multi-robot collaborative complex indoor scene segmentation via multiplex interactive learning, *CAAI Transactions on Intelligence Technology*, vol. 10, no. 6, pp. 1646–1660, 2025.

[231] Z. Mandi, S. Jain, and S. Song, RoCo: Dialectic multi-robot collaboration with large language models, in *IEEE International Conference on Robotics and Automation*, 2024, pp. 286–299.

[232] S. S. Kannan, V. L. Venkatesh, and B.-C. Min, Smart-LLM: Smart multi-agent robot task planning using large language models, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2024, pp. 12 140–12 147.

[233] Y. Lan, X. Xu, J. Liu, X. Zhang, Y. Lu, and L. Cheng, A survey on reinforcement learning for optimal decision-making and control of intelligent vehicles, *CAAI Transactions on Intelligence Technology*, vol. 10, no. 6, pp. 1593–1615, 2025.

[234] Q. Lu *et al.*, Realistic corner case generation for autonomous vehicles with multimodal large language model, *Tsinghua Science and Technology*, vol. 31, no. 3, pp. 1440–1459, 2026.

[235] J.-J. Hwang *et al.*, EMMA: End-to-end multimodal model for autonomous driving, *Transactions on Machine Learning Research*, 2025. [Online]. Available: https://openreview.net/forum?id=kH3t5lmOU8

[236] S. Xing *et al.*, OpenEMMA: Open-source multimodal model for end-to-end autonomous driving, in *Winter Conference on Applications of Computer Vision*, 2025, pp. 1001–1009.

[237] Z. Zhou *et al.*, AutoVLA: A vision-language-action model for end-to-end autonomous driving with adaptive reasoning and reinforcement fine-tuning, in *Advances in Neural Information Processing Systems*, vol. 38, 2025, pp. 27 920–27 956.

[238] H. Fu *et al.*, ORION: A holistic end-to-end autonomous driving framework by vision-language instructed action generation, in *IEEE/CVF International Conference on Computer Vision*, 2025, pp. 24 823–24 834.

[239] Y. Li *et al.*, ReCogDrive: A reinforced cognitive frame-

work for end-to-end autonomous driving, *arXiv preprint arXiv:2506.08052*, 2025.

[240] S. Zhang, W. Huang, Z. Gao, H. Chen, and C. Lv, WiseAD: Knowledge augmented end-to-end autonomous driving with vision-language model, *arXiv preprint arXiv:2412.09951*, 2024.

[241] B. Jiang, S. Chen, Q. Zhang, W. Liu, and X. Wang, AlphaDrive: Unleashing the power of VLMs in autonomous driving via reinforcement learning and reasoning, *arXiv preprint arXiv:2503.07608*, 2025.

[242] Z. Huang *et al.*, Gen-Drive: Enhancing diffusion generative driving policies with reward modeling and reinforcement learning fine-tuning, in *IEEE International Conference on Robotics and Automation*, 2025, pp. 3445–3451.

[243] R. Zhao *et al.*, Sce2DriveX: A generalized MLLM framework for scene-to-drive learning, *IEEE Robotics and Automation Letters*, vol. 10, no. 12, pp. 12 580–12 587, 2025.

[244] S. Wang *et al.*, OmniDrive: A holistic vision-language dataset for autonomous driving with counterfactual reasoning, in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 22 442–22 452.

[245] D. Rivkin, F. Hogan, A. Feriani, A. Konar, A. Sigal, S. Liu, and G. Dudek, SAGE: smart home agent with grounded execution, *arXiv preprint arXiv:2311.00772*, 2023.

[246] S. Li, Y. Guo, J. Yao, Z. Liu, and H. Wang, HomeBench: Evaluating LLMs in smart homes with valid and invalid instructions across single and multiple devices, in *Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 12 230–12 250.

[247] H. Cui, Y. Du, Q. Yang, Y. Shao, and S. C. Liew, LLMind: Orchestrating AI and IoT with LLM for complex task execution, *IEEE Communications Magazine*, vol. 63, no. 4, pp. 214–220, 2025.

[248] L. Shen, Q. Yang, Y. Zheng, and M. Li, AutoIOT: LLM-driven automated natural language programming for AIoT applications, in *Proceedings of the 31st Annual International Conference on Mobile Computing and Networking*, 2025, p. 468–482.

[249] J. Lee, J. Wang, E. Brown, L. Chu, S. S. Rodriguez, and J. E. Froehlich, GazePointAR: A context-aware multimodal voice assistant for pronoun disambiguation in wearable augmented reality, in *CHI Conference on Human Factors in Computing Systems*, 2024.

[250] C. Zhu, S.-K. Hsia, X. Hu, Z. Liu, J. Shi, and K. Ramani, AgentAR: Creating augmented reality applications with tool-augmented LLM-based autonomous agents, 2025.

[251] M. Schuck, D. O. Dahanaggamaarachchi, B. Sprenger, V. Vyas, S. Zhou, and A. P. Schoellig, SwarmGPT: Combining large language models with safe motion planning for drone swarm choreography, *IEEE Robotics and Automation Letters*, vol. 10, no. 11, pp. 12 237–12 244, 2025.

[252] O. Sautenkov *et al.*, UAV-VLA: Vision-language-action system for large scale aerial mission generation, in *2025 20th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2025, pp. 1588–1592.

[253] A. Lykov, S. Karaf, M. Martynov, V. Serpiva, A. Fedoseev, M. Konenkov, and D. Tsetserukou, FlockGPT: Guiding UAV flocking with linguistic orchestration, in *2024 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, 2024, pp. 485–488.

[254] J. Zhao and X. Lin, General-purpose aerial intelligent agents empowered by large language models, *arXiv preprint arXiv:2503.08302*, 2025.

[255] J. Li *et al.*, The dawn after the dark: An empirical study on factuality hallucination in large language models, in *Association for Computational Linguistics*, 2024, pp. 10 879–10 899.

[256] X.-W. Yang *et al.*, Neuro-symbolic artificial intelligence: Towards improving the reasoning abilities of large language models, in *International Joint Conference on Artificial Intelligence*, 2025, pp. 10 770–10 778.

[257] A. P. Smit, N. Grinsztajn, P. Duckworth, T. D. Barrett, and A. Pretorius, Should we be going MAD? a look at multi-agent debate strategies for LLMs, in *International Conference on Machine Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=CrUmgUaAQp

[258] Z. Zhou, J. Song, K. Yao, Z. Shu, and L. Ma, ISR-LLM: Iterative self-refined large language model for long-horizon sequential task planning, in *IEEE International Conference on Robotics and Automation*, 2024, pp. 2081–2088.

[259] Z. Zhang and F. Liu, Cost-augmented Monte Carlo Tree Search for LLM-assisted planning, *arXiv preprint arXiv:2505.14656*, 2025.

[260] L. Guan, K. Valmeekam, S. Sreedharan, and S. Kambhampati, Leveraging pre-trained large language models to construct and utilize world models for model-based task planning, in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 79 081–79 094.

[261] W. Fang, Y. Zhang, K. Qian, J. R. Glass, and Y. Zhu, PLAY2PROMPT: Zero-shot tool instruction optimization for LLM agents via tool play, in *Findings of the Association for Computational Linguistics: ACL 2025*, 2025, pp. 26 274–26 290.

[262] C. Qin, M. Liu, and L. Jin, Adaptive gradient activation for federated learning on consumer electronic devices, *IEEE Transactions on Consumer Electronics*, vol. 71, no. 2, pp. 5489–5500, 2025.

[263] A. Mannekote, A. Davies, J. Kang, and K. E. Boyer, Can LLMs reliably simulate human learner actions? a simulation authoring framework for open-ended learning environments, in *AAAI Conference on Artificial Intelligence*, vol. 39, no. 28, 2025, pp. 29 044–29 052.

[264] A. Razdaibiedina, Y. Mao, R. Hou, M. Khabsa, M. Lewis, and A. Almahairi, Progressive prompts: Continual learning for language models, in *International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=UJTgQBc91_

[265] Y. He, E. Wang, Y. Rong, Z. Cheng, and H. Chen, Security of AI agents, in *IEEE/ACM International Workshop on Responsible AI Engineering*, 2025, pp. 45–52.

[266] S. Vijayvargiya *et al.*, OpenAgentSafety: A comprehensive framework for evaluating real-world AI agent safety, in *International Conference on Learning Representations*, 2026. [Online]. Available: https://openreview.net/forum?id=xggSxCFQbA

[267] Z. Zeng, J. Wang, and X. Liao, Stability analysis of delayed cellular neural networks described using cloning templates, *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 51, no. 11, pp. 2313–2324, 2004.

[268] H. Su, J. Zhang, and Z. Zeng, Formation-containment control of multi-robot systems under a stochastic sampling mechanism, *Science China Technological Sciences*, vol. 63, no. 6, pp. 1025–1034, 2020.

[269] D. Ma, Y. Guan, L. Jin, and S. Li, Distributed optimal control of multiple serial robot systems with kinematics and dynamics based on discrete neural dynamics, *IEEE Transactions on Industrial Electronics*, vol. 72, no. 6, pp. 6393–6401, 2025.

[270] D. Ma, J. Lv, C. Xu, and L. Jin, Logic-adaptive discrete neural dynamics for distributed cooperative control of multi-robot systems via minimum infinity norm optimization, *IEEE Transactions on Fuzzy Systems*, vol. 34, no. 2, pp. 531–539, 2026.

[271] S. Han, Q. Zhang, Y. Yao, W. Jin, and Z. Xu, LLM multi-agent systems: Challenges and open problems, *arXiv preprint arXiv:2402.03578*, 2024.

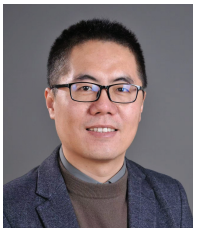
[272] H. Li *et al.*, Theory of mind for multi-agent collaboration via large language models, in *Conference on Empirical Methods in Natural Language Processing*, 2023, pp. 180–192.

[273] Q. Long, F. Zhong, M. Wu, Y. Wang, and S.-C. Zhu, SocialGFs: Learning social gradient fields for multi-agent reinforcement learning, *arXiv preprint arXiv:2405.01839*, 2024.

[274] W. Chen, G. Zhang, J. Wu, and G. Jeon, Experimental evidence on the emergent social network patterns of LLM-driven multi-agent system for edge service collaboration, *Tsinghua Science and Technology*, 2026.

Author biography

Chuan Qin received the B.E. degree from Harbin Institute of Technology at Weihai, Weihai, China, in 2021. He is currently pursuing a Ph.D. degree in computer science and technology with the School of Information Science and Engineering, Lanzhou University, Lanzhou, China. His current research interests include large language models and optimization.



Long Jin received the B.E. degree in automation and the Ph.D. degree in information and communication engineering from Sun Yat-sen University, Guangzhou, China, in 2011 and 2016, respectively.

From 2016 to 2017, he underwent postdoctoral training with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong. Since 2017, he has been a Professor with Lanzhou University, Lanzhou, China. He is also a Chair Professor with Lanzhou University of Technology, with the approval of both Lanzhou University and Lanzhou University of Technology. His research interests include neural networks, optimization, intelligent computing, and robotics.

Prof. Jin is currently an Associate Editor for several journals, such as IEEE Transactions on Industrial Electronics, IEEE/CAA Journal of Automatica Sinica, and IEEE Transactions on Automation Science and Engineering.