A Survey on Large Language Model-Based Autonomous Agents

Anonymous Authors Anonymous Institution
City, Country
Email: anonymous@example.com

Abstract—The emergence of Large Language Models (LLMs) has catalyzed a paradigm shift in autonomous agent research, enabling a new generation of systems that leverage the knowledge and reasoning capabilities embedded within these models to tackle complex tasks across diverse domains. This survey provides a comprehensive examination of LLM-based autonomous agents, organizing the rapidly growing body of work through the Brain-Perception-Action (BPA) framework—a unified taxonomy that conceptualizes agent architectures through three fundamental modules: the Brain Module for reasoning, planning, and decision-making; the Perception Module for environmental sensing and state representation; and the Action Module for tool use, output generation, and physical interaction.

We systematically review over 150 papers spanning the period from 2022 to early 2025, covering architectural innovations in planning systems (Chain-of-Thought, Tree-of-Thought, Graph-of-Thought reasoning), memory systems (working memory, episodic memory, procedural memory), tool use and function calling, and multi-agent collaboration patterns. The survey examines applications across six major domains—code development, web navigation, gaming, robotics and embodied AI, scientific research, and personal assistants—analyzing domain-specific adaptations and cross-domain patterns.

We further provide a systematic analysis of evaluation methodologies, benchmarks, and metrics, identifying critical gaps between benchmark performance and real-world deployment reliability. Safety and alignment considerations are examined throughout, reflecting their critical importance as agents become increasingly capable and autonomous. Finally, we identify key open challenges including long-horizon planning, robust memory systems, reliable tool use, and scalable multi-agent coordination, while outlining promising research directions for the field.

This survey serves as both a comprehensive reference for researchers entering the field and a roadmap for future development of more capable, reliable, and safe autonomous agents.

Index Terms—Large Language Models, Autonomous Agents, Artificial Intelligence, Agent Architecture, Multi-Agent Systems, Tool Learning, Planning, Memory Systems.

1 Introduction

1.1 Motivation

The pursuit of autonomous agents—artificial systems capable of perceiving their environment, reasoning about goals, and taking actions to achieve those goals without continuous human oversight—has been a central theme in artificial intelligence research since its inception [Russell and Norvig(2020)]. From early symbolic agents operating in constrained domains to modern reinforcement learning systems mastering complex games, the field has witnessed remarkable progress. However, a fundamental challenge has persisted: how to build agents that possess sufficient knowledge, reasoning capability, and adaptability to operate effectively across diverse, open-ended environments.

The emergence of Large Language Models (LLMs) has catalyzed a paradigm shift in autonomous agent research. Models such as GPT-4, Claude, and LLaMA, trained on vast corpora of text spanning virtually every domain of human knowledge, have demonstrated unprecedented capabilities in natural language understanding, generation, and reasoning [Brown et al.(2020)]. This development has given rise to a new generation of *LLM-based autonomous agents* that leverage the knowledge and reasoning abilities embedded within these models to tackle complex tasks that were previously intractable for traditional agent architectures.

The significance of this development cannot be overstated. Unlike traditional agents that require extensive domain-specific training or hand-crafted knowledge bases, LLM-based agents can be rapidly deployed to novel tasks through natural language instructions, exhibit emergent reasoning capabilities, and adapt to new situations through in-context learning. This flexibility has sparked an explosion of research activity, with hundreds of papers published in the past two years exploring various aspects of LLM-based agents, from architectural innovations to application domains.

The rapid pace of development, while exciting, has created challenges for researchers and practitioners attempting to navigate this evolving landscape. The diversity of approaches, the lack of standardized terminology, and the absence of a unified framework for comparing different systems have made it difficult to identify key advances, understand their relationships, and determine promising directions for future research. This survey aims to address these challenges by providing a comprehensive, systematic review of the field.

1.2 Problem Definition

Before proceeding, it is essential to establish a clear definition of what constitutes an autonomous agent in the context of this survey. Drawing from both classical AI literature and

recent developments, we define an LLM-based autonomous agent as follows:

[LLM-based Autonomous Agent] An LLM-based autonomous agent is a computational system that: (1) utilizes a Large Language Model as its core reasoning engine; (2) can perceive information from its environment through various modalities; (3) can take actions to affect change in that environment, including but not limited to tool use, code execution, and physical actuation; (4) operates with some degree of autonomy in pursuing specified goals; and (5) can adapt its behavior based on feedback and accumulated experience.

This definition distinguishes LLM-based agents from several related concepts. Unlike standalone LLMs that simply generate text in response to prompts, autonomous agents maintain persistent state, interact with external environments, and execute multi-step plans. Unlike traditional software agents with fixed behavioral rules, LLM-based agents can reason about novel situations and adapt their strategies accordingly. Unlike pure reinforcement learning agents, they leverage the vast knowledge encoded in pre-trained language models, enabling them to operate effectively with minimal task-specific training.

The architecture of LLM-based agents can be understood through the *Brain-Perception-Action (BPA)* framework, which organizes agent capabilities into three fundamental modules [Wang et al.(2023b)], [Sumers et al.(2024)]:

- **Brain Module**: The central cognitive processing unit responsible for reasoning, planning, memory management, and decision-making. This module leverages the LLM's capabilities for chain-of-thought reasoning [Wei et al.(2022a)], tree-of-thought exploration [Yao et al.(2023b)], and other sophisticated reasoning patterns.
- **Perception Module**: The interface through which the agent acquires information from its environment. This includes textual perception (natural language understanding), visual perception (image and video analysis), auditory perception, and structured data perception (databases, knowledge graphs).
- **Action Module**: The mechanisms through which the agent affects change in its environment. This encompasses tool use and API invocation [Schick et al.(2023)], code generation and execution, natural language output, and in embodied systems, physical actions such as navigation and manipulation.

This framework provides a principled basis for analyzing and comparing different agent architectures, identifying component-level innovations, and understanding the design space of autonomous agents.

1.3 Scope of Survey

This survey provides a comprehensive review of LLM-based autonomous agents, covering the period from 2022 to early 2025, with particular emphasis on works published since the introduction of foundational frameworks such as ReAct [Yao et al.(2023d)] and Chain-of-Thought prompting [Wei et al.(2022a)]. The scope is delineated along several dimensions:

Technical Scope: We focus on agents that use LLMs as their primary reasoning engine, including both single-agent systems and multi-agent collaborations. We cover the full spectrum of agent capabilities including planning, memory, tool use, and multi-agent coordination. We exclude purely reinforcement learning-based agents that do not incorporate LLM reasoning, as well as traditional symbolic agents that predate the LLM era.

Application Domains: We examine applications across diverse domains including software engineering (code agents), web navigation, game playing, embodied AI and robotics, scientific discovery, healthcare, finance, and personal assistance. This breadth reflects the versatility of LLM-based agents and enables cross-domain insights.

Evaluation and Safety: We dedicate substantial attention to evaluation methodologies, benchmarks, and safety considerations, recognizing these as critical factors for the responsible development and deployment of autonomous agents [Ma et al.(2025)].

Exclusions: We do not provide detailed coverage of LLM training methodologies, prompt engineering techniques divorced from agent applications, or multi-agent reinforcement learning systems that do not incorporate LLM-based reasoning. These topics, while important, are better addressed in dedicated surveys.

The survey is informed by a systematic literature review process. We searched major academic databases (arXiv, ACL Anthology, NeurIPS Proceedings, ICLR Open Review) using keywords including "autonomous agents," "LLM agents," "language agents," "multi-agent systems," and "embodied AI." We prioritized highly-cited papers, papers from top-tier venues, and papers introducing significant methodological innovations. Our final corpus includes over 150 papers spanning the full range of topics covered in this survey.

1.4 Contributions

This survey makes the following contributions to the field:

1) **Unified Framework**: We present the Brain-Perception-Action (BPA) taxonomy as a unified framework for understanding, comparing, and developing LLM-based autonomous agents. This framework synthesizes insights from cognitive architectures [Sumers et al.(2024)], multi-agent systems literature, and recent advances in LLM-based agents to provide systematic organization of the field.
2) **Comprehensive Taxonomy**: We develop a detailed hierarchical taxonomy covering all major components of autonomous agents, including planning systems (task decomposition, reasoning methods, plan optimization), memory systems (sensory, working, and long-term memory), perception mechanisms, action execution (tool use, output generation, physical actions), and multi-agent collaboration patterns.
3) **Systematic Literature Review**: We provide a comprehensive review of over 150 papers, organizing them according to our taxonomy and identifying key trends, innovations, and relationships among

different approaches. This addresses the fragmentation in current literature and provides researchers with a coherent map of the field.

4) **Cross-Domain Analysis**: We analyze agent applications across multiple domains, identifying common patterns, domain-specific adaptations, and opportunities for cross-pollination of ideas. This enables researchers from different application areas to benefit from innovations in other domains.
5) **Evaluation Landscape**: We survey the rapidly evolving landscape of agent evaluation, covering benchmarks, evaluation frameworks, and metrics. We identify gaps in current evaluation practices and propose directions for more comprehensive assessment.
6) **Safety and Alignment**: We dedicate attention to safety considerations, reviewing threats, defense mechanisms, and alignment approaches specific to autonomous agents. This reflects the critical importance of safety as agents become more capable and autonomous.
7) **Future Directions**: We identify key open challenges and promising research directions across all aspects of autonomous agents, from fundamental architectural questions to practical deployment considerations.

1.5 Relation to Existing Surveys

Several surveys have addressed aspects of LLM-based agents. Wang et al. [Wang et al.(2023b)] provided an early comprehensive survey focusing on agent construction, applications, and evaluation. Sumers et al. [Sumers et al.(2024)] proposed the CoALA framework for understanding language agents through the lens of cognitive architectures. Liu et al. [Liu et al.(2024a)] surveyed embodied AI with emphasis on multi-modal large models. Ma et al. [Ma et al.(2025)] provided an extensive review of safety considerations for large models and agents.

This survey complements and extends these works in several ways. First, we provide a more comprehensive and up-to-date coverage of the field, incorporating numerous papers published after previous surveys. Second, our BPA taxonomy offers a different organizational perspective that emphasizes the functional decomposition of agent capabilities. Third, we provide more extensive coverage of multi-agent systems, an area of growing importance. Fourth, we offer deeper analysis of the relationships between different approaches and clearer identification of research gaps. Finally, we integrate safety and evaluation considerations throughout the survey rather than treating them as separate topics.

1.6 Paper Organization

The remainder of this survey is organized as follows:

Section 2 provides background on the historical development of autonomous agents, foundational LLM capabilities that enable agent applications, and key concepts from multi-agent systems that inform current research.

Section 3 presents our Brain-Perception-Action taxonomy in detail, covering planning and reasoning systems, memory architectures, tool use mechanisms, and multi-agent collaboration patterns within a unified framework.

Section 4 surveys application domains, including code agents, web navigation, game playing, embodied AI, scientific discovery, healthcare, finance, and personal assistants.

Section 5 covers evaluation methodologies, benchmarks, and metrics for assessing agent capabilities across different dimensions, including safety considerations.

Section 6 identifies open challenges and future research directions across all aspects of autonomous agents.

Section 7 concludes with a summary of key insights and a vision for the future of autonomous agents.

Throughout the survey, we use consistent terminology and notation, provide cross-references between related topics, and highlight connections between different sections. We aim to make this survey accessible to readers from diverse backgrounds while maintaining the technical depth expected by domain experts.

2 Background

This section provides the foundational context for understanding LLM-based autonomous agents. We trace the historical evolution of agent systems from early symbolic approaches to modern neural architectures, introduce the fundamentals of large language models, and discuss the paradigm shift that has enabled the emergence of autonomous agents powered by LLMs.

2.1 The Evolution of Agent Systems

The concept of an autonomous agent—a computational system capable of perceiving its environment and taking actions to achieve specified goals—has been central to artificial intelligence research since its inception. The evolution of agent architectures reflects broader shifts in AI paradigms, from symbolic reasoning to statistical learning and, most recently, to neural approaches grounded in large-scale pre-training.

2.1.1 Symbolic AI and Classical Agent Architectures

The earliest agent systems emerged from the symbolic AI tradition, also known as "Good Old-Fashioned AI" (GOFAI), which dominated the field from the 1950s through the 1980s. These systems operated on the principle that intelligence could be realized through the manipulation of symbolic representations using explicit rules and logical inference mechanisms. Agent architectures from this era, such as expert systems and planning agents, relied on handcrafted knowledge bases and reasoning engines that could derive conclusions from explicitly represented facts and rules.

A significant advancement in classical agent architecture came with the development of cognitive architectures such as SOAR [Laird et al.(1987)] and ACT-R [Anderson and Lebiere(1996)], which provided integrated frameworks for modeling human cognition. These architectures incorporated modules for declarative memory, procedural knowledge, and goal-directed reasoning, establishing design patterns that continue to influence modern agent systems. The Belief-Desire-Intention (BDI) architecture [Rao

and Georgeff(1995)] further advanced agent design by providing a formal framework for practical reasoning, enabling agents to maintain mental attitudes and select actions based on their beliefs about the world, their desires, and their intentions.

However, symbolic agent systems faced fundamental limitations. The "knowledge acquisition bottleneck"—the challenge of manually encoding the vast knowledge required for general intelligence—proved insurmountable for many real-world applications. Furthermore, these systems lacked robustness in the face of uncertainty and could not easily adapt to novel situations outside their programmed knowledge.

2.1.2 Reinforcement Learning Agents

The limitations of symbolic approaches motivated the development of learning-based agent architectures, particularly those grounded in reinforcement learning (RL). In the RL paradigm, agents learn optimal behavior through interaction with an environment, receiving feedback in the form of rewards or penalties. This approach enabled agents to acquire complex behaviors without explicit programming, demonstrating remarkable success in domains ranging from game playing [Silver et al.(2016)] to robotic control [Levine et al.(2018)].

Multi-agent systems research extended these ideas to settings where multiple agents interact, studying coordination, communication, and competition [Albrecht and Stone(2018)]. Deep reinforcement learning, combining RL with neural network function approximation, further expanded the capabilities of learning-based agents, achieving superhuman performance in complex games such as Go and StarCraft.

Despite these successes, RL-based agents faced their own challenges. They required extensive interaction with their environments to learn effective policies, struggled with sparse reward settings, and had difficulty generalizing beyond their training distributions. The sample inefficiency of RL methods and their reliance on carefully designed reward functions limited their applicability to many real-world scenarios.

2.1.3 The Neural-Symbolic Transition

The emergence of large language models has catalyzed a new paradigm in agent design that bridges neural and symbolic approaches. Unlike traditional RL agents that learn from environment interaction, LLM-based agents leverage knowledge acquired from massive text corpora during pretraining. This knowledge includes not only factual information but also procedural knowledge about how to approach problems, reason through complex tasks, and interact with tools and systems.

Figure 1 illustrates the key milestones in the development of LLM-based autonomous agents from 2022 to 2025.

This transition represents a fundamental shift in how agents acquire and deploy knowledge. Rather than learning behaviors from scratch through environmental interaction, LLM-based agents begin with a rich foundation of world knowledge and reasoning capabilities, which can be directed toward specific tasks through prompting and fine-tuning. This approach addresses many limitations of both symbolic and RL-based agents: it eliminates the knowledge acquisition bottleneck, provides strong generalization capabilities, and enables rapid adaptation to new tasks without extensive training.

2.2 Large Language Model Fundamentals

Large Language Models form the cognitive foundation of modern autonomous agents. Understanding their architecture, capabilities, and limitations is essential for comprehending how they can be integrated into agent systems.

2.2.1 Transformer Architecture

The transformer architecture, introduced by Vaswani et al. with the landmark "Attention Is All You Need" paper, fundamentally transformed natural language processing and enabled the development of modern LLMs. The key innovation of the transformer is the self-attention mechanism, which allows the model to weigh the importance of different parts of the input when producing each element of the output. This parallelizable architecture replaced the sequential processing of recurrent neural networks, enabling training on much larger datasets.

The transformer consists of an encoder and decoder, each composed of stacked layers of self-attention and feed-forward networks. Modern LLMs typically use decoder-only variants, which are trained to predict the next token in a sequence given the preceding context. This autoregressive training objective, combined with massive scale—both in terms of parameters and training data—has proven remarkably effective at inducing general language understanding and generation capabilities.

2.2.2 Pretraining and Scaling

The development of the GPT (Generative Pre-trained Transformer) series demonstrated the power of pretraining on large text corpora. GPT-2 showed that a language model trained on diverse web text could generate coherent, contextually appropriate continuations across a wide range of topics. GPT-3 [Brown et al.(2020)] dramatically scaled this approach to 175 billion parameters, revealing that sufficiently large models exhibit emergent capabilities not present in smaller models.

A crucial discovery was that large language models can perform tasks through few-shot or zero-shot learning—that is, by providing examples or instructions in the prompt without updating model parameters. This in-context learning capability enables LLMs to adapt to new tasks at inference time, a property that is fundamental to their use in agent systems where flexibility and rapid adaptation are essential.

The scaling laws governing language model performance [Kaplan et al.(2020)] established predictable relationships between model size, dataset size, and performance, motivating continued scaling. Models such as GPT-4, Claude, and LLaMA have pushed the boundaries of scale and capability, demonstrating increasingly sophisticated reasoning, knowledge synthesis, and instruction-following abilities.

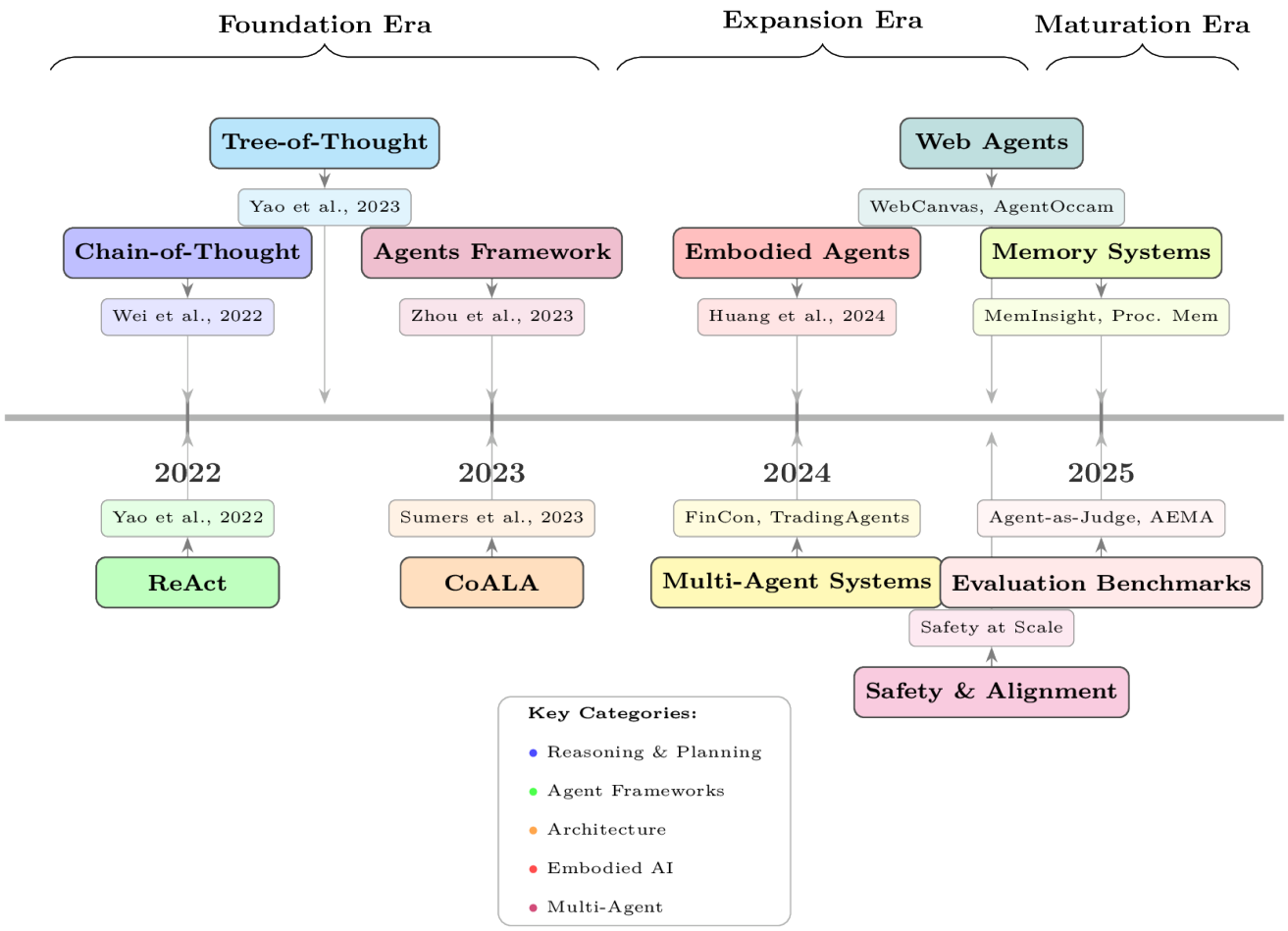

Fig. 1: Timeline of key developments in LLM-based autonomous agents from 2022 to 2025. The Foundation Era established core reasoning frameworks. The Expansion Era saw rapid growth in agent architectures and multi-agent systems. The Maturation Era focuses on sophisticated memory systems and evaluation benchmarks.

2.2.3 Reasoning Capabilities and Chain-of-Thought

A significant advancement in leveraging LLMs for complex tasks came with the development of chain-of-thought (CoT) prompting [Wei et al.(2022a)]. Wei et al. demonstrated that prompting language models to generate intermediate reasoning steps before producing a final answer significantly improves performance on arithmetic, commonsense, and symbolic reasoning tasks. This finding revealed that LLMs possess latent reasoning capabilities that can be elicited through appropriate prompting strategies.

The Tree of Thoughts (ToT) framework [Yao et al.(2023b)] extended this approach by enabling exploration over multiple reasoning paths, allowing models to deliberate, backtrack, and evaluate different approaches to problem-solving. These reasoning frameworks are foundational to agent systems, which must often plan, reason about goals, and make decisions in complex environments.

2.3 From Language Models to Autonomous Agents

The transition from LLMs as static language processors to dynamic autonomous agents represents a paradigm shift in AI system design. This transformation has been enabled by key methodological innovations that bridge the gap between language understanding and goal-directed action.

2.3.1 Reasoning and Acting: The ReAct Paradigm

The ReAct (Reasoning and Acting) framework [Yao et al.(2023d)] marked a pivotal moment in the development of LLM-based agents. Yao et al. demonstrated that interleaving reasoning traces with task-specific actions enables LLMs to interact with external environments while maintaining coherent goal-directed behavior. In the ReAct paradigm, the model generates thoughts that help track progress, plan next steps, and handle exceptions, while actions allow it to query external knowledge sources or interact with environments.

This synergistic combination addresses key limitations of pure reasoning approaches: reasoning traces help the model induce, track, and update action plans, while actions provide grounding in external information and environments. ReAct demonstrated significant improvements on question answering, fact verification, and interactive decision-making tasks, establishing a template for subsequent agent architectures.

2.3.2 Tool Use and Function Calling

The ability to use external tools significantly extends the capabilities of LLM-based agents beyond their parametric knowledge. Tool use enables agents to access up-to-date information, perform precise computations, and interact with external systems and APIs. This capability transforms LLMs from isolated language processors into components of larger systems that can effect change in the world.

Tool-augmented agents must address several challenges: determining when tool use is necessary, selecting appropriate tools from available options, constructing correct tool invocations, and integrating tool outputs into ongoing reasoning. Various approaches have been developed to address these challenges, including retrieval-based tool selection, learning-based methods, and structured prompting strategies that guide models through tool-use workflows.

2.3.3 Memory and Long-Term Reasoning

Effective autonomous agents require memory systems that extend beyond the limited context windows of language models. Drawing inspiration from cognitive science, researchers have developed memory architectures that distinguish between working memory (active information being processed), episodic memory (records of past experiences), and semantic memory (general knowledge).

Memory-augmented agents can store and retrieve relevant experiences, learn from past interactions, and maintain consistent behavior over extended time periods. These capabilities are essential for tasks requiring long-term planning, personalization, or accumulation of knowledge through experience. The integration of retrieval mechanisms with LLMs has proven particularly effective, enabling agents to access relevant information from large knowledge bases during reasoning.

2.3.4 Multi-Agent Systems

The agent paradigm naturally extends to settings where multiple specialized agents collaborate to achieve complex goals. Multi-agent systems built on LLMs can decompose tasks among agents with different roles, enable peer review and quality control, and implement hierarchical control structures with manager and worker agents.

These systems draw on both the classical multi-agent systems literature and the cognitive architectures tradition, combining explicit coordination mechanisms with the flexible reasoning capabilities of LLMs. Applications range from software development teams to scientific discovery systems, demonstrating the power of collaborative agent architectures.

2.3.5 A Unified Framework: Brain-Perception-Action

The Brain-Perception-Action (BPA) framework [Wang et al.(2023a)], [Sumers et al.(2024)] provides a unified perspective for understanding LLM-based autonomous agents. In this framework:

- **Brain**: The LLM serves as the central cognitive processing unit, responsible for reasoning, planning, memory management, and decision-making. This module draws on the emergent capabilities of large language models to process information, generate plans, and select actions.
- **Perception**: Input modules acquire and process information from the environment, transforming raw sensory inputs (text, images, structured data) into representations suitable for cognitive processing. Perception enables the agent to understand its current state and relevant context.
- **Action**: Output modules execute decisions through tool use, natural language generation, code execution, or physical actuation. The action module bridges the gap between the agent's internal reasoning and external effects.

This framework synthesizes insights from cognitive science with the practical requirements of building autonomous systems, providing both a conceptual lens for understanding existing work and a design pattern for developing new agent architectures. The modular structure enables systematic improvement of individual components while maintaining overall system coherence.

The emergence of LLM-based autonomous agents represents not merely an incremental advance but a fundamental reconceptualization of how intelligent systems can be built. By leveraging the knowledge and reasoning capabilities acquired through large-scale pretraining, these systems address longstanding challenges in agent design while opening new possibilities for human-AI collaboration and autonomous problem-solving.

3 Agent Architecture

The architecture of an autonomous agent defines its structural organization, determining how it perceives the environment, processes information, makes decisions, and executes actions. Drawing inspiration from cognitive science and biological systems, we organize our discussion around the **Brain-Perception-Action (BPA)** paradigm, which provides a unified framework for understanding and comparing diverse agent architectures [Sumers et al.(2024)]. This paradigm aligns with the Cognitive Architectures for Language Agents (CoALA) framework, which emphasizes modular memory components, structured action spaces, and generalized decision-making processes.

3.1 Overview of the BPA Framework

The Brain-Perception-Action framework conceptualizes autonomous agents through three fundamental modules that work in concert to enable intelligent behavior:

- **Brain Module**: Serves as the central cognitive processing unit, responsible for reasoning, planning, memory management, and decision-making. This module acts analogously to the human prefrontal cortex, orchestrating high-level cognitive functions.
- **Perception Module**: Handles information acquisition and processing from the environment, transforming raw sensory inputs into structured representations suitable for cognitive processing. This parallels biological sensory systems such as vision and hearing.

- **Action Module**: Executes decisions through tool use, output generation, and physical interactions, enabling the agent to affect changes in its environment. This corresponds to motor systems and tool manipulation capabilities in biological agents.

These three modules are interconnected through information flows that form a perception-cognition-action loop, enabling agents to interact with their environments in a goal-directed manner. Figure 2 illustrates the overall BPA framework architecture.

The following sections examine each module in detail, along with the critical dimension of multi-agent collaboration that emerges when multiple agents coordinate their activities.

3.2 Brain Module: Planning and Reasoning

The Brain Module constitutes the cognitive core of an autonomous agent, encompassing the computational processes that enable planning, reasoning, and decision-making. This module determines how agents decompose complex tasks, select appropriate strategies, and adapt their behavior based on feedback.

3.2.1 Planning Systems

Planning systems generate sequences of actions to achieve specified goals while considering constraints and available resources. We identify three primary approaches to planning in LLM-based agents:

3.2.1.1 Task Decomposition: Complex tasks often require decomposition into manageable sub-tasks. The ReAct framework [Yao et al.(2023d)] pioneered an interleaved approach where reasoning traces and task-specific actions alternate, enabling agents to induce, track, and update action plans dynamically. This synergy between reasoning and acting allows agents to handle exceptions and gather additional information through environment interaction.

Task decomposition strategies can be categorized along a spectrum from single-step to hierarchical planning:

- **Single-step Planning**: Generate one action at a time based on current state, providing flexibility but requiring more inference calls.
- **Multi-step Sequential Planning**: Generate complete action sequences before execution, enabling optimization of the overall plan but reducing adaptability.
- **Hierarchical Task Networks (HTN)**: Employ multi-level decomposition with abstract and concrete plans, enabling reasoning at multiple levels of abstraction. Systems like S-Agents [Chen et al.(2024)] demonstrate the effectiveness of self-organizing hierarchical structures in open-ended environments.

3.2.1.2 Reasoning Methods: The quality of planning depends critically on the underlying reasoning capabilities. Several paradigms have emerged for enhancing LLM reasoning:

Chain-of-Thought (CoT) prompting [Wei et al.(2022b)] elicits reasoning in large language models by generating intermediate reasoning steps. This simple yet powerful approach significantly improves performance on arithmetic, commonsense, and symbolic reasoning tasks by making the reasoning process explicit.

Tree-of-Thought (ToT) [Yao et al.(2023c)] extends CoT by exploring multiple reasoning paths in a tree structure. ToT enables deliberate decision-making through consideration of different reasoning paths, self-evaluation of choices, and backtracking when necessary. On challenging tasks such as Game of 24, ToT achieves substantially higher success rates compared to standard CoT prompting.

Graph-of-Thought (GoT) [Yao et al.(2023a)] further generalizes reasoning by representing thoughts as arbitrary graph structures rather than linear chains or trees. This enables non-linear reasoning patterns including aggregation of multiple reasoning paths and transformation of intermediate results.

Buffer of Thoughts (BoT) [Yang et al.(2024c)] introduces a meta-buffer that stores informative high-level thought templates distilled from problem-solving processes across various tasks. By retrieving and adapting relevant templates, BoT achieves significant performance improvements while requiring only a fraction of the computational cost of multi-query prompting methods.

3.2.1.3 Plan Optimization and Self-Correction: Effective agents must refine their plans based on execution feedback. Reflexion [Shinn et al.(2023)] introduces verbal reinforcement learning, where agents reflect on task feedback signals and maintain reflective text in episodic memory buffers to induce better decision-making in subsequent trials. This approach achieves substantial improvements across sequential decision-making, coding, and language reasoning tasks.

Self-Refine [Madaan et al.(2023)] demonstrates that LLMs can iteratively improve their outputs through self-generated feedback and refinement, without requiring additional training or reinforcement learning. This iterative process improves task performance by approximately 20% on average across diverse tasks.

Learning from Failure [Wang et al.(2024)] shows that unsuccessful trajectories offer valuable insights for agent fine-tuning. By incorporating negative examples with appropriate quality control, agents can achieve better performance on mathematical reasoning, multi-hop question answering, and strategic reasoning tasks.

3.2.2 Memory Systems

Memory systems enable agents to store, organize, and retrieve information, supporting learning and adaptation over time. Drawing from cognitive psychology, we distinguish three primary memory types with distinct temporal characteristics and functional roles:

3.2.2.1 Sensory Memory: Sensory memory provides brief retention of perceptual information, typically lasting milliseconds to seconds. In agent architectures, this manifests as:

- Visual buffers for processing screenshots or camera feeds
- Auditory buffers for speech recognition and audio processing
- Temporary storage for multi-modal inputs before integration

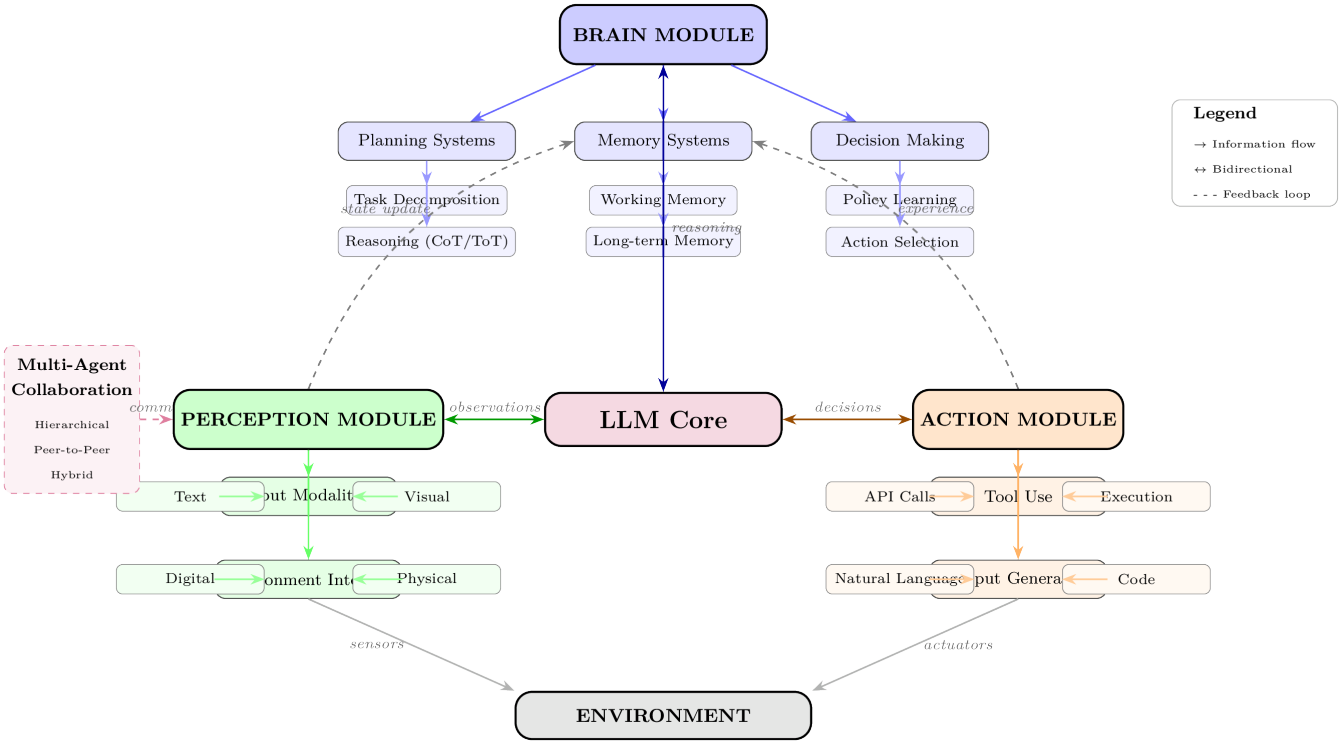

Fig. 2: The Brain-Perception-Action (BPA) framework for autonomous agents. The Brain Module handles planning, memory, and decision-making. The Perception Module processes multi-modal inputs. The Action Module executes through tool use. The central LLM Core coordinates information flow.

3.2.2.2 Working Memory: Working memory maintains and manipulates information actively during task execution, with capacity limited by the model's context window. Key considerations include:

Context Window Management: The fixed context window of LLMs constrains the amount of information available for reasoning. MemGPT [Packer et al.(2023)] introduces virtual context management, drawing inspiration from hierarchical memory systems in operating systems to provide the appearance of large memory resources through data movement between fast and slow memory tiers.

KV Cache Optimization: Efficient management of key-value caches enables longer effective context and reduced computational costs. Recent work on persistent KV caches [Shkolnikov(2026)] demonstrates methods for maintaining memory state across extended interactions.

3.2.2.3 Long-term Memory: Long-term memory provides permanent storage for experiences, knowledge, and skills. We distinguish three subtypes:

Episodic Memory stores personal experiences and interaction trajectories. Generative Agents [Park et al.(2023)] demonstrate how episodic memories can be synthesized into higher-level reflections and retrieved dynamically to plan behavior, enabling believable simulations of human behavior in interactive environments.

Semantic Memory encodes facts, concepts, and world knowledge. This can be implemented through:

- Vector databases for semantic search and retrieval
- Knowledge graphs for structured knowledge representation
- Parametric knowledge encoded in model weights

Procedural Memory stores skills and "how-to" knowledge. Voyager [Wang et al.(2023c)] demonstrates an ever-growing skill library of executable code for storing and retrieving complex behaviors, enabling rapid capability compounding in open-ended environments.

3.2.2.4 Memory Operations: Memory systems must support several key operations:

- **Encoding**: Transforming experiences into memory representations through vector embeddings or structured storage formats.
- **Retrieval**: Accessing stored information through semantic search, associative recall, or temporal indexing. Embodied-RAG [Xie et al.(2024)] demonstrates non-parametric embodied memory for physical environments.
- **Consolidation**: Stabilizing and integrating memories through experience replay, knowledge distillation, or reflection processes.
- **Forgetting**: Removing outdated or irrelevant information through decay mechanisms or importance filtering to prevent memory overflow and maintain relevance.

3.2.3 Decision Making

Decision-making processes select actions from available options based on goals, constraints, and environmental state. We identify three key aspects:

3.2.3.1 Policy Learning: Agents can acquire decision-making policies through various mechanisms:

- **In-context Learning**: Acquire behaviors from examples provided in prompts, enabling rapid adaptation without parameter updates.
- **Imitation Learning**: Learn from demonstrations of desired behavior, either through fine-tuning or prompt engineering.
- **Reinforcement Learning from Feedback**: Optimize policies based on reward signals from environment feedback or human preferences.

3.2.3.2 Action Selection: Given a policy, agents must select specific actions:

- **Greedy Selection**: Choose the highest-utility action deterministically.
- **Probabilistic Sampling**: Sample from action distributions, enabling exploration and diversity.
- **Constrained Optimization**: Select actions that maximize utility while satisfying constraints.

3.2.3.3 Uncertainty Handling: Robust decision-making requires handling uncertainty:

- **Confidence Estimation**: Assess certainty in predictions and decisions.
- **Exploration-Exploitation Trade-off**: Balance gathering new information with exploiting known strategies.
- **Risk-aware Decision Making**: Incorporate risk estimates into action selection.

3.3 Perception Module: Input Processing

The Perception Module acquires and processes information from the environment, transforming raw sensory inputs into structured representations suitable for cognitive processing. This module determines the agent's ability to understand and respond to its surroundings.

3.3.1 Input Modalities

Modern agents process diverse input types, each requiring specialized processing:

3.3.1.1 Textual Perception: Text remains the primary modality for LLM-based agents, encompassing:

- Natural language understanding for user instructions and communications
- Document analysis for processing reports, articles, and structured text
- Instruction parsing for extracting actionable information from commands
- Multi-turn dialogue understanding for conversational interactions

3.3.1.2 Visual Perception: Visual capabilities extend agents' reach into domains requiring scene understanding:

- Image and video understanding for content analysis
- Screenshot analysis for web agents navigating graphical interfaces
- Spatial reasoning from visual input for embodied agents
- Object detection and tracking for robotics applications

WebVoyager [He et al.(2024)] demonstrates how large multimodal models can process both visual and textual information to navigate real-world websites end-to-end, achieving significantly higher success rates compared to text-only approaches.

3.3.1.3 Structured Data Perception: Agents must interface with structured information sources:

- Database schema understanding for query formulation
- Knowledge graph traversal for reasoning over structured knowledge
- API response interpretation for tool integration
- Tabular data analysis for numerical reasoning

3.3.2 Environment Interfaces

Perception modules must adapt to diverse operational environments:

3.3.2.1 Digital Environments:

- **Web Browsers**: Navigate and interact with web pages through DOM parsing, screenshot analysis, and element interaction. WebCanvas [Pan et al.(2024)] provides an online evaluation framework that captures the dynamic nature of web interactions.
- **Operating Systems**: File system operations, process management, and system-level interactions.
- **Mobile Devices**: App navigation and touch interactions, as demonstrated by AutoDroid [Wen et al.(2023)].
- **Code Repositories**: Code understanding, version control, and software development workflows.

3.3.2.2 Physical Environments:

- **Robotic Platforms**: Manipulation and navigation in physical space through sensor integration.
- **Autonomous Vehicles**: Driving and traffic navigation through multi-modal perception.
- **IoT Devices**: Smart home and industrial automation through sensor networks.

3.3.2.3 Virtual Environments:

- **Game Worlds**: Minecraft [Wang et al.(2023c)], NetHack, and other game environments provide rich testbeds for agent development.
- **Simulation Platforms**: Physics simulators and social simulations enable safe testing and training.

3.3.3 State Representation

Effective perception requires constructing meaningful state representations:

3.3.3.1 Internal State:

- **Belief States**: The agent's model of the world, including uncertainties and probabilities.
- **Goal Representations**: Current objectives, sub-goals, and their relationships.
- **Task Progress**: Completion status, milestones achieved, and remaining work.

3.3.3.2 External State:

- **Environment Observations**: Current perceptual inputs from sensors and interfaces.
- **Multi-agent Awareness**: Other agents' states, intentions, and predicted behaviors.
- **Temporal Evolution**: Historical state sequences and trends over time.

3.4 Action Module: Tool Use and Execution

The Action Module executes decisions through tool use, output generation, and physical interactions, enabling agents to affect changes in their environment.

3.4.1 Tool Use

Tool use encompasses the selection, invocation, and management of external capabilities that extend the agent's native abilities.

3.4.1.1 Tool Categories: We classify tools by their primary function:

- **Information Retrieval Tools**: Search engines, database queries, and knowledge bases for gathering information.
- **Computation Tools**: Code execution, mathematical solvers, and simulators for performing calculations.
- **Communication Tools**: Messaging APIs, email systems, and social media for interacting with humans and other agents.
- **Domain-Specific Tools**: Medical diagnosis systems, financial analysis tools, and scientific instruments for specialized applications.

3.4.1.2 Tool Learning and Selection: Toolformer [Schick et al.(2023)] demonstrates that language models can learn to use external tools through self-supervised learning, deciding which APIs to call, when to call them, and how to incorporate results into future predictions.

Tool selection mechanisms include:

- **Retrieval-based Selection**: Select tools from a knowledge base using semantic similarity between task descriptions and tool capabilities.
- **Learning-based Selection**: Train models to predict optimal tool selection based on task characteristics.
- **Cost-aware Planning**: Consider execution costs, latency, and resource requirements in selection decisions.

3.4.1.3 Tool Integration Frameworks: HuggingGPT [Shen et al.(2023)] presents an LLM-powered controller that connects various AI models to solve complex tasks. The system uses ChatGPT for task planning, model selection based on function descriptions, execution with selected models, and response summarization.

3.4.2 Output Generation

Agents produce various output types depending on task requirements:

3.4.2.1 Natural Language Output:

- Response generation for user interaction
- Explanation synthesis for transparency and interpretability
- Report and summary writing for documentation

3.4.2.2 Code Generation: Chain of Code [Li et al.(2024a)] demonstrates how code-writing can enhance reasoning by providing a structured medium for complex computations. The approach combines code execution with an "LMulator" that simulates execution for semantic subtasks.

Code generation capabilities include:

- Program synthesis from natural language specifications
- Script generation for automation tasks
- Query construction for databases and APIs

3.4.2.3 Structured Output:

- JSON/XML generation for API interactions
- Database operations (CRUD)
- Configuration file generation

3.4.3 Physical Actions (Embodied Agents)

For embodied agents, the Action Module must generate physical actions:

3.4.3.1 Navigation Actions:

- Path planning and execution
- Obstacle avoidance
- Terrain adaptation

3.4.3.2 Manipulation Actions:

- Grasping and object interaction
- Tool manipulation
- Precise motor control

Embodied Tree of Thoughts [Xu et al.(2025b)] demonstrates how tree search through action spaces, guided by physics-based simulation, can enable deliberate manipulation planning in robotic systems.

3.5 Multi-Agent Collaboration

Multi-agent collaboration involves coordination, communication, and cooperation among multiple autonomous agents to achieve individual or collective goals. This dimension has emerged as a powerful paradigm for tackling complex tasks that benefit from specialization and parallel processing.

3.5.1 Organization Structures

The organizational structure defines how agents relate to one another:

3.5.1.1 Hierarchical Structures: In hierarchical organizations, agents are arranged in layers with control flowing from higher to lower levels:

- **Manager-Worker Architecture**: Manager agents handle planning, coordination, and result integration, while worker agents execute specialized tasks.
- **Layered Control**: Multiple levels of abstraction with higher layers providing strategic guidance and lower layers handling tactical execution.
- **Recursive Decomposition**: Tasks are recursively broken down and delegated to specialized sub-agents.

MetaGPT [Hong et al.(2024)] encodes Standardized Operating Procedures (SOPs) into prompt sequences, assigning diverse roles to agents in an assembly line paradigm. This approach generates more coherent solutions than chat-based multi-agent systems by enabling verification of intermediate results.

3.5.1.2 Flat/Peer-to-Peer Structures: In flat organizations, agents participate as equals:

- **Equal Participation**: All agents contribute equally to problem-solving.
- **Voting Mechanisms**: Decisions made through collective voting procedures.
- **Consensus Building**: Agents negotiate to reach agreement on actions.

3.5.1.3 Hybrid Structures: Hybrid organizations combine elements of hierarchical and flat structures:

- Dynamic role assignment based on task requirements
- Context-dependent hierarchies that reconfigure as needed
- Fluid team structures that adapt to changing conditions

3.5.2 Communication Protocols

Effective multi-agent systems require well-defined communication mechanisms:

3.5.2.1 Message Types:

- **Task Assignments**: Delegate work to specific agents with clear specifications.
- **Status Updates**: Report progress, completion, or issues encountered.
- **Query/Response**: Request and provide information between agents.
- **Negotiation Messages**: Resolve conflicts, allocate resources, and coordinate actions.

3.5.2.2 Communication Patterns:

- **Broadcast**: One-to-all communication for global announcements or shared context.
- **Point-to-Point**: Direct agent-to-agent communication for private coordination.
- **Publish-Subscribe**: Topic-based distribution for event-driven coordination.
- **Blackboard Systems**: Shared information space for collaborative problem solving.

3.5.3 Collaboration Paradigms

Multi-agent interactions can be characterized by the alignment of agent objectives:

3.5.3.1 Cooperative Collaboration: Agents share common goals and work together toward collective success:

- Shared objectives with distributed task execution
- Joint problem solving with complementary capabilities
- Examples include FinCon [Yu et al.(2024)] for financial decision-making and MedAide [Yang et al.(2024b)] for medical diagnosis

3.5.3.2 Competitive Interactions: Agents have opposing objectives in zero-sum or adversarial settings:

- Zero-sum games where one agent's gain is another's loss
- Adversarial scenarios requiring strategic reasoning
- Market-based competition in economic simulations

3.5.3.3 Mixed-Motive Scenarios: Agents have partially aligned interests requiring negotiation:

- Coalition formation for achieving shared sub-goals
- Bargaining over resources or outcomes
- Complex social and economic interactions

3.5.4 Role Specialization

Specialized roles enable division of labor and expertise development:

3.5.4.1 Functional Roles:

- **Planner Agents**: Task decomposition, plan generation, and strategy formulation.
- **Executor Agents**: Action execution, tool invocation, and plan implementation.
- **Evaluator Agents**: Result assessment, quality control, and feedback generation.
- **Coordinator Agents**: Communication management, conflict resolution, and resource allocation.

3.5.4.2 Domain Roles:

- **Domain Experts**: Specialized knowledge in specific fields (medicine, finance, law).
- **Tool Specialists**: Expert in particular tool categories or APIs.
- **Memory Managers**: Information storage, retrieval, and organization.
- **Safety Monitors**: Constraint enforcement and risk assessment.

3.6 Architectural Integration

The effective integration of Brain, Perception, and Action modules determines overall agent capability. Key integration challenges include:

3.6.0.1 Information Flow Management: Information must flow seamlessly between modules:

- Perception-to-Brain: Structured representations from sensory inputs
- Brain-to-Action: Executable plans and commands
- Action-to-Perception: Feedback and environment updates

3.6.0.2 Temporal Coordination: Modules must operate with appropriate timing:

- Real-time perception for dynamic environments
- Deliberative reasoning for complex decisions
- Timely action execution for task completion

3.6.0.3 Resource Allocation: Computational and memory resources must be distributed appropriately:

- Balancing perception bandwidth with reasoning depth
- Managing working memory for multi-step tasks
- Optimizing action selection for efficiency

3.7 Summary

The Brain-Perception-Action framework provides a systematic lens for understanding autonomous agent architectures. The Brain Module enables sophisticated planning and reasoning through methods like Chain-of-Thought, Tree-of-Thought, and iterative refinement. The Perception Module processes multi-modal inputs from diverse environments, constructing meaningful state representations. The Action Module extends agent capabilities through tool use and generates appropriate outputs. Multi-agent collaboration adds a crucial dimension for complex task solving through specialized roles and coordinated communication.

This architectural perspective reveals both the accomplishments of current systems and the challenges that remain. The following sections examine how these architectural components are applied across diverse domains and the evaluation methodologies used to assess agent capabilities.

4 Applications

The versatility of LLM-based autonomous agents has enabled their deployment across a diverse range of domains, each presenting unique challenges and opportunities. This section examines six major application areas where autonomous agents have demonstrated significant impact: code development, web navigation, gaming, robotics, scientific research, and personal assistants. For each domain, we analyze the specific requirements, representative systems, and current capabilities.

4.1 Code Development

Software development represents one of the most successful application domains for autonomous agents, leveraging the natural alignment between programming languages and the structured reasoning capabilities of large language models. Code agents assist developers across the software development lifecycle, from initial code generation to debugging, testing, and maintenance.

4.1.1 Code Generation and Synthesis

Code generation agents transform natural language specifications into executable programs, bridging the gap between human intent and machine execution. WizardCoder [Luo et al.(2023)] demonstrates how evol-instruct methodologies can empower code LLMs to achieve competitive performance on code generation benchmarks. The system iteratively refines code solutions through evolutionary approaches, significantly improving code quality and correctness.

Multi-agent architectures have emerged as particularly effective for complex software engineering tasks. The Multi-LLM Agent framework [Shen et al.(2024)] decomposes the tool-use capabilities into specialized components: a planner for task decomposition, a caller for tool invocation, and a summarizer for result synthesis. This modular approach enables smaller language models to collaborate effectively, achieving performance competitive with larger monolithic models while maintaining computational efficiency.

Context engineering approaches have further advanced code assistant capabilities. Recent work [Haseeb(2025)] demonstrates how combining intent clarification, retrieval-augmented generation, and specialized sub-agents orchestrated through multi-agent frameworks can significantly improve accuracy and reliability in real-world repositories. The system achieves higher single-shot success rates and better adherence to project context compared to baseline single-agent approaches.

4.1.2 Software Engineering Automation

Beyond code generation, autonomous agents increasingly support comprehensive software engineering workflows. Agent-driven automatic software improvement [Ruiz(2024)] explores how agents can autonomously identify, implement, and verify software improvements. These systems analyze codebases, detect potential enhancements, and generate patches while maintaining code quality and test coverage.

The vision of agentic software engineering [Hoda(2025)] extends beyond code generation to encompass the entire development process, including requirements analysis, architecture design, implementation, testing, and deployment. Agents in this paradigm serve as collaborative partners that augment human developers' capabilities rather than replacing them entirely.

4.1.3 Challenges and Opportunities

Code agents face several persistent challenges. Long-context management remains critical for understanding large codebases, requiring efficient memory mechanisms to maintain relevant context across files and modules. Hallucination in code generation can lead to syntactically correct but semantically incorrect programs, necessitating robust verification mechanisms. Additionally, the security implications of generated code require careful consideration, as agents may inadvertently introduce vulnerabilities.

4.2 Web Navigation

Web navigation agents autonomously interact with websites to accomplish user-specified goals, from filling forms to gathering information and completing transactions. This domain presents unique challenges due to the dynamic, heterogeneous, and often unpredictable nature of web environments.

4.2.1 Browser Automation Agents

WebCanvas [Pan et al.(2024)] introduces an innovative online evaluation framework that addresses the dynamic nature of web interactions. Unlike static benchmarks, WebCanvas evaluates agents in live web environments, capturing the continuously evolving nature of real-world websites. The framework introduces novel evaluation metrics that identify critical intermediate actions necessary for task completion while filtering noise from insignificant events. On the Mind2Web-Live benchmark containing 542 tasks with 2,439 intermediate evaluation states, the best-performing agents achieve approximately 23% task success rate and 49% task completion rate.

AgentOccam [Yang et al.(2024a)] presents a simple yet strong baseline for LLM-based web agents, demonstrating

that careful architectural decisions can significantly impact performance. The system emphasizes the importance of hybrid context management combining accessibility tree snapshots with selective vision capabilities, achieving approximately 85% success rate on the WebGames benchmark.

4.2.2 Multi-Modal Web Understanding

Modern web agents must process both visual and textual information to navigate complex websites effectively. WebVoyager demonstrates how large multimodal models can process screenshots alongside textual content to navigate real-world websites end-to-end. The visual modality proves essential for understanding page layouts, identifying interactive elements, and detecting dynamic content that may not be captured in the DOM structure alone.

Web-CogReasoner [Guo et al.(2025b)] introduces a knowledge-driven cognitive reasoning framework that decomposes web agent capabilities into knowledge content learning and cognitive processes. The system categorizes knowledge into factual, conceptual, and procedural types, enabling agents to acquire the domain knowledge necessary for effective web interaction. This approach demonstrates significant improvements in generalizing to unseen tasks where structured knowledge proves decisive.

4.2.3 Security Considerations

Web agents face significant security challenges. Research on user-mediated attacks [Chen et al.(2026)] reveals that commercial agents often bypass safety constraints in over 92% of cases when processing unverified content, converting such content into confident booking guidance. Web-use agents exhibit near-deterministic execution of risky actions, highlighting the need for robust safety mechanisms that prioritize security alongside helpfulness.

4.3 Gaming

Game environments provide rich testbeds for autonomous agent research, offering diverse challenges including strategic reasoning, multi-agent coordination, exploration, and skill acquisition. The complexity of game worlds, combined with clear success metrics, makes gaming an ideal domain for developing and evaluating agent capabilities.

4.3.1 Open-World Game Agents

Minecraft has emerged as a premier environment for developing open-ended learning agents. Odyssey [Liu et al.(2024b)] empowers LLM-based agents with an open-world skill library comprising 40 primitive skills and 183 compositional skills, enabling exploration of the vast Minecraft world. The framework includes a fine-tuned LLaMA-3 model trained on 390k+ instruction entries derived from the Minecraft Wiki, demonstrating effective evaluation of agent capabilities across long-term planning, dynamic-immediate planning, and autonomous exploration tasks.

Voyager represents a landmark achievement in open-ended learning, demonstrating how agents can continuously expand their capabilities through self-driven exploration. The system maintains an ever-growing skill library of executable code for storing and retrieving complex behaviors, enabling rapid capability compounding in open-ended environments.

4.3.2 Strategic Game Playing

AvalonBench evaluates LLMs playing the game of Avalon, a social deduction game requiring sophisticated reasoning about other players' intentions and beliefs. The benchmark reveals both the potential and limitations of current agents in strategic social reasoning, highlighting challenges in modeling other agents' mental states.

ALYMPICS [Mao et al.(2023b)] explores the intersection of LLM agents and game theory, providing a framework for evaluating strategic decision-making in competitive and cooperative settings. The system demonstrates how agents can reason about game-theoretic concepts such as Nash equilibrium and dominant strategies, though performance varies significantly across game types.

Research on regret in LLM agents [Park et al.(2024)] examines whether agents can learn from past mistakes in online learning scenarios. The findings reveal that while agents exhibit some capacity for regret minimization, significant gaps remain compared to optimal game-theoretic strategies.

4.3.3 Multi-Agent Game Coordination

Game environments increasingly serve as testbeds for multi-agent coordination. Research on Hanabi demonstrates how augmenting action spaces with conventions [Bredell et al.(2025)] can significantly improve multi-agent cooperation. These conventions, based on human gameplay patterns, enable implicit knowledge sharing that enhances coordination under partial observability.

PillagerBench extends Minecraft-based evaluation to competitive scenarios, benchmarking LLM-based agents in adversarial settings where agents must balance resource gathering with combat and defense.

4.4 Robotics and Embodied AI

Embodied agents bridge the gap between digital intelligence and physical action, enabling autonomous systems to perceive, reason about, and interact with the physical world. This domain presents unique challenges including real-time perception, safe physical interaction, and robust execution under uncertainty.

4.4.1 Embodied Generalist Agents

An Embodied Generalist Agent in 3D World [Huang et al.(2023)] demonstrates how agents can develop diverse capabilities for operating in three-dimensional environments. The system integrates perception, navigation, and manipulation skills, enabling agents to accomplish a wide range of tasks in simulated 3D environments.

Embodied-RAG [Xie et al.(2024)] introduces non-parametric embodied memory for physical environments, enabling agents to retrieve and leverage past experiences when navigating novel situations. This approach proves particularly valuable for long-horizon tasks where agents must maintain consistent behavior over extended periods.

4.4.2 Robotic Control and Manipulation

Robotic Control via Embodied Chain-of-Thought Reasoning [Zawalski et al.(2024)] demonstrates how deliberate reasoning can enhance robotic manipulation. The system applies

chain-of-thought prompting to generate action sequences, enabling more robust and interpretable robot behavior.

Research on LLM-based agents for embodied robot cognition [Shaji et al.(2026)] investigates the extent to which LLMs can serve as core components for planning and execution reasoning in cognitive robot architectures. The proposed architecture uses an agentic LLM as the central component for planning and reasoning, supported by working and episodic memories for learning from experience. Evaluation on household tasks demonstrates that LLM-driven agents can complete structured tasks and exhibit emergent adaptation, though significant limitations remain including hallucinations about task success.

4.4.3 Autonomous Driving

A Language Agent for Autonomous Driving [Mao et al.(2023a)] demonstrates how LLM-based agents can enhance decision-making in autonomous vehicles. The system processes natural language commands and environmental descriptions to generate driving decisions, enabling more intuitive human-vehicle interaction and adaptable behavior in novel situations.

4.4.4 Challenges in Embodied AI

Embodied agents face unique challenges including the need for real-time perception and action, safety constraints in physical environments, and the sim-to-real transfer problem. BadRobot [Zhang et al.(2024b)] investigates jailbreaking attacks on embodied LLMs, revealing vulnerabilities that emerge when language models control physical systems. These findings underscore the importance of robust safety mechanisms for embodied agents operating in the physical world.

4.5 Scientific Research

Autonomous agents increasingly contribute to scientific discovery, assisting researchers across the scientific method from hypothesis generation to experimental design, data analysis, and literature synthesis.

4.5.1 Automated Machine Learning

AutoML-Agent [Trirat et al.(2024)] presents a multi-agent LLM framework for full-pipeline automated machine learning. The system decomposes the ML workflow across specialized agents handling data preprocessing, feature engineering, model selection, hyperparameter optimization, and evaluation. This multi-agent approach enables comprehensive automation of the ML pipeline while maintaining flexibility for domain-specific customization.

4.5.2 Materials Discovery

Hierarchical multi-agent LLM reasoning for autonomous functional materials discovery [Rothfarb et al.(2025)] demonstrates how agents can accelerate scientific research in materials science. The system employs a hierarchical organization of agents, with higher-level agents coordinating strategy and lower-level agents executing specific computational and experimental tasks. This architecture enables efficient exploration of the materials design space while maintaining scientific rigor.

4.5.3 Knowledge Synthesis

Knowledge-driven agentic scientific corpus distillation frameworks [Xiao et al.(2025)] address the challenge of processing the exponentially growing scientific literature. These systems autonomously identify, extract, and synthesize relevant knowledge from scientific publications, enabling researchers to maintain awareness of developments across their fields.

LLM-SRBench [Shojaee et al.(2025)] provides a benchmark for evaluating LLM agents on scientific equation discovery, assessing their ability to identify mathematical relationships from experimental data. The benchmark reveals both the potential for accelerating scientific discovery and the limitations of current systems in capturing complex scientific phenomena.

4.5.4 Domain-Specific Applications

Agents have demonstrated success across scientific domains. In healthcare, MedAide [Yang et al.(2024b)] introduces an LLM-based medical multi-agent collaboration framework that enables intent-aware information fusion across specialized healthcare domains. The system combines syntactic constraints with retrieval-augmented generation to decompose complex medical queries, achieving improvements in medical proficiency and strategic reasoning.

In finance, TradingAgents [Xiao et al.(2024)] proposes a novel stock trading framework featuring LLM-powered agents in specialized roles including fundamental analysts, sentiment analysts, technical analysts, and traders. The framework simulates collaborative trading environments, demonstrating improvements in cumulative returns, Sharpe ratio, and maximum drawdown compared to baseline models.

4.6 Personal Assistants

Personal assistant agents represent perhaps the most visible application of autonomous agent technology, directly serving end users in daily tasks ranging from scheduling and communication to information retrieval and task automation.

4.6.1 Task Automation

Personal assistant agents excel at automating routine tasks that would otherwise require significant user effort. Research on plan reuse mechanisms [Li et al.(2024b)] demonstrates that approximately 30% of requests received by LLM-driven agents are identical or similar, enabling significant latency reduction through intelligent plan caching and reuse. The AgentReuse mechanism achieves 93% effective plan reuse rate, reducing latency by over 93% compared to baselines without reuse mechanisms.

AutoDroid [Wen et al.(2023)] brings LLM-powered task automation to Android devices, enabling agents to navigate apps, interact with UI elements, and complete multi-step tasks on mobile platforms. The system demonstrates the feasibility of deploying autonomous agents on resource-constrained mobile devices while maintaining task completion accuracy.

4.6.2 Multi-Device Coordination

Modern personal assistants must coordinate across multiple devices and platforms. Agents increasingly manage IoT devices, coordinate with smart home systems, and synchronize activities across smartphones, tablets, and computers. This multi-device coordination requires sophisticated state management and consistent behavior across heterogeneous platforms.

4.6.3 Conversational Assistants

The conversational interface remains central to personal assistant interaction. Generative Agents [Park et al.(2023)] demonstrates how agents can maintain consistent personalities, remember past interactions, and engage in believable social interactions. The system stores agent experiences in natural language, synthesizes memories into higher-level reflections, and retrieves them dynamically to plan behavior, enabling 25 agents to operate coherently in an interactive sandbox environment.

Focus Agent [Zhang et al.(2024d)] explores LLM-powered virtual focus groups, demonstrating how agents can simulate group discussions and act as moderators. This application highlights the potential for agents to serve not just as individual assistants but as facilitators of collective activities.

4.6.4 Challenges in Personal Assistant Deployment

Personal assistants face unique challenges including privacy protection, personalization, and long-term relationship building. The intimate nature of personal assistant interactions requires careful attention to data handling and user trust. Additionally, assistants must adapt to individual user preferences while maintaining consistent behavior, requiring sophisticated personalization mechanisms that balance customization with reliability.

4.7 Cross-Domain Analysis

Examining applications across domains reveals common patterns and domain-specific adaptations. Table 1 summarizes the key characteristics of each application domain.

4.7.1 Common Architectural Patterns

Across domains, several architectural patterns emerge as particularly effective. Multi-agent collaboration proves valuable when tasks require diverse expertise or can benefit from parallel processing. Memory systems, both episodic and semantic, enable agents to maintain context and learn from experience. Tool use extends agent capabilities beyond their native abilities, enabling interaction with external systems and APIs.

4.7.2 Domain-Specific Adaptations

Each domain requires specific adaptations of general agent architectures. Code agents emphasize precise syntax generation and execution verification. Web agents require robust handling of dynamic content and multi-modal perception. Game agents need strategic reasoning and exploration capabilities. Embodied agents must integrate perception with physical action under real-time constraints. Scientific agents require domain knowledge integration and rigorous validation. Personal assistants prioritize natural interaction and personalization.

TABLE 1: Summary of Autonomous Agent Applications Across Domains

Domain	Primary Challenges	Key Capabilities	Repres System
Code Development	Long-context management, hallucination, security	Code generation, debugging, testing	Wizard LLM A
Web Navigation	Dynamic environments, multi-modal perception, security	Browser automation, form filling, information extraction	WebCa cam
Gaming	Strategic reasoning, exploration, multi-agent coordination	Skill acquisition, game playing, social reasoning	Odysse
Robotics	Real-time execution, safety, sim-to-real transfer	Navigation, manipulation, perception	Embod Robotic
Scientific Research	Domain knowledge, experimental design, validation	Hypothesis generation, data analysis, literature synthesis	AutoM MedAi
Personal Assistants	Privacy, personalization, long-term interaction	Task automation, scheduling, conversational interaction	AutoD Agents

The continued development of autonomous agents across these diverse applications demonstrates the versatility of the LLM-based agent paradigm while highlighting areas requiring further research. The following section examines evaluation methodologies for assessing agent capabilities across these varied domains.

5 Evaluation Methods

The evaluation of autonomous agents presents unique challenges that distinguish it from traditional machine learning assessment. Unlike static models that produce single outputs, agents engage in multi-step interactions with dynamic environments, making evaluation more complex and multi-faceted. This section examines the benchmarks, metrics, and evaluation frameworks developed to assess agent capabilities across diverse domains.

5.1 Evaluation Challenges and Dimensions

The evaluation of LLM-based autonomous agents requires consideration of multiple dimensions that extend beyond traditional NLP metrics. We identify five key evaluation dimensions:

Task Performance measures the agent's ability to achieve specified goals, including success rate, completion quality, and goal satisfaction. Unlike single-turn tasks, agent tasks often involve multiple steps with intermediate objectives that must be tracked and evaluated.

Process Quality assesses how the agent achieves its goals, including the efficiency of action sequences, the quality of reasoning traces, and the appropriateness of tool selections. This dimension recognizes that optimal outcomes achieved through suboptimal processes may not generalize.

Robustness evaluates the agent's resilience to perturbations, including adversarial inputs, distribution shifts,

and environmental changes. Agents operating in real-world environments must handle unexpected situations gracefully.

Safety examines whether the agent avoids harmful behaviors, respects constraints, and operates within acceptable risk boundaries. This dimension is particularly critical for agents with real-world action capabilities.

Generalization measures the agent's ability to transfer learned capabilities to novel tasks, environments, and tool combinations. Strong agents should demonstrate zero-shot and few-shot adaptation beyond their training distribution.

5.2 Benchmark Categories

The proliferation of agent systems has motivated the development of numerous benchmarks tailored to specific domains and capabilities. We organize these benchmarks into six primary categories based on the agent's operational domain. Table 2 provides a summary of major evaluation benchmarks.

5.2.1 General Agent Benchmarks

General agent benchmarks evaluate broad capabilities across multiple task types. Agent-as-a-Judge [Zhuge et al.(2024)] introduces a framework wherein agentic systems evaluate other agentic systems, extending the LLM-as-a-Judge paradigm to incorporate agentic features that enable intermediate feedback throughout the task-solving process. The accompanying DevAI benchmark comprises 55 realistic AI development tasks with 365 hierarchical user requirements, demonstrating that agentic evaluation achieves reliability comparable to human evaluation while dramatically reducing manual effort.

MAFBench [Orogat et al.(2026)] provides a unified evaluation suite for multi-agent LLM frameworks, integrating existing benchmarks under a standardized execution pipeline. This framework enables controlled comparison of architectural choices, revealing that framework-level design decisions alone can induce order-of-magnitude differences in latency, reduce planning accuracy by up to 30%, and lower coordination success from above 90% to below 30%.

5.2.2 Web Agent Benchmarks

Web agents navigate and interact with web interfaces, requiring capabilities in visual understanding, DOM manipulation, and multi-step task completion. WebCanvas [Pan et al.(2024)] addresses the dynamic nature of web environments through an online evaluation framework that captures critical intermediate states while disregarding noise from insignificant events. The associated Mind2Web-Live dataset contains 542 tasks with 2,439 intermediate evaluation states, enabling realistic assessment of web agent performance in evolving environments.

AgentDAM [Zharmagambetov et al.(2025)] introduces a benchmark for evaluating privacy preservation in web-navigation agents, measuring adherence to the data minimization principle. The benchmark reveals that current agents frequently process unnecessary sensitive information, highlighting the gap between capability and appropriate information handling.

SecureWebArena [Ying et al.(2025)] provides a holistic security evaluation benchmark for vision-language model-based web agents, encompassing 2,970 trajectories across six simulated web environments and defining a structured taxonomy of six attack vectors spanning user-level and environment-level manipulations.

5.2.3 Embodied Agent Benchmarks

Embodied agents operate in physical or simulated 3D environments, requiring integration of perception, planning, and motor control. EmbodiedCity [Gao et al.(2024)] constructs a benchmark platform for embodied intelligence evaluation in real-world city environments, combining a highly realistic 3D simulation environment with high-fidelity pedestrian and vehicle flow simulations.

Embodied Arena [Ni et al.(2025)] establishes a comprehensive evaluation platform with a systematic capability taxonomy spanning perception, reasoning, and task execution across 25 fine-grained dimensions. The platform integrates 22 diverse benchmarks across three domains: 2D/3D Embodied Q&A, Navigation, and Task Planning.

BEDI [Guo et al.(2025a)] provides a systematic benchmark for UAV-Embodied Agents, introducing a Dynamic Chain-of-Embodied-Task paradigm that decomposes complex tasks into standardized subtasks across six core skills: semantic perception, spatial perception, motion control, tool utilization, task planning, and action generation.

5.2.4 Game Playing Benchmarks

Game environments provide controlled yet challenging testbeds for agent evaluation. AvalonBench [Light et al.(2023)] evaluates LLM agents playing the strategic social deduction game Resistance Avalon, where players must deceive, deduce, and negotiate. The benchmark reveals significant capability gaps: ChatGPT playing good roles achieves only a 22.2% win rate against rule-based bots playing evil roles, compared to 38.2% for good-role bots in the same setting.

Orak [Park et al.(2025)] presents a foundational benchmark spanning 12 popular video games across all major genres, enabling comprehensive studies of LLM capabilities essential for intricate game scenarios. The benchmark introduces a plug-and-play interface based on Model Context Protocol and includes fine-tuning datasets for aligning pre-trained LLMs into gaming agents.

5.2.5 Code and Software Engineering Benchmarks

Code agents require evaluation of program synthesis, debugging, and software development capabilities. PPTC [Guo et al.(2023)] and PPTC-R [Zhang et al.(2024c)] evaluate LLMs on PowerPoint task completion, measuring robustness to adversarial user instructions at sentence, semantic, and multi-language levels, as well as adaptation to software version changes.

5.2.6 Multi-Agent Coordination Benchmarks

Multi-agent systems require evaluation of coordination, communication, and collaboration capabilities. ZSC-Eval [Wang et al.(2023d)] provides the first evaluation toolkit for zero-shot coordination algorithms, introducing generation of evaluation partner candidates through behavior-preferring rewards and measuring generalization performance via Best-Response Proximity metrics.

TABLE 2: Summary of Major Evaluation Benchmarks for Autonomous Agents

Benchmark	Focus Area	Metrics	Year
Agent-as-a-Judge	General capability	Task success, Quality	2024
WebCanvas	Web navigation	Task completion, Steps	2024
AvalonBench	Social deduction	Win rate, Detection	2023
EmbodiedCity	Embodied agents	Navigation success	2024
ZSC-Eval	Multi-agent coordination	Zero-shot performance	2023
PPTC/PPTC-R	Office tasks	Task completion	2023-24
AEMA	Trustworthiness	Verifiable evaluation	2024
GeoBenchX	Geospatial tasks	Multi-step accuracy	2025
MAFBench	Multi-agent frameworks	Latency, Coordination	2026

5.3 Evaluation Metrics

The complexity of agent behavior necessitates diverse metrics that capture different aspects of performance.

5.3.1 Task-Level Metrics

Success Rate measures the proportion of tasks completed successfully according to predefined criteria. This fundamental metric provides a coarse-grained assessment of overall capability.

Task Completion Rate captures partial success, measuring the degree to which sub-goals are achieved even when the overall task fails. WebCanvas [Pan et al.(2024)] demonstrates that distinguishing between complete failure and partial progress provides more informative evaluation.

Step Efficiency evaluates the number of actions required to complete tasks, penalizing unnecessarily lengthy solutions while rewarding efficient planning.

5.3.2 Process-Level Metrics

Action Validity Rate measures the proportion of generated actions that are executable within the environment, capturing the agent's understanding of action constraints.

Tool Selection Accuracy assesses the appropriateness of tool choices given task requirements and context. ToolRet [Shi et al.(2025)] reveals that even strong information retrieval models exhibit poor performance on tool retrieval tasks, with low retrieval quality significantly degrading task pass rates.

Reasoning Quality evaluates the logical coherence and completeness of the agent's reasoning traces, often assessed through human evaluation or LLM-as-a-Judge approaches.

5.3.3 Safety Metrics

Constraint Satisfaction Rate measures adherence to specified constraints and safety boundaries. Agent-SafetyBench [Zhang et al.(2024a)] reveals that none of 16 evaluated LLM agents achieves a safety score above 60%, highlighting significant safety challenges.

Risk Awareness Score assesses the agent's ability to identify and appropriately respond to potentially risky situations. R-Judge [Yuan et al.(2024)] benchmarks this capability across 569 multi-turn interaction records encompassing 27 key risk scenarios.

Harm Prevention Rate measures the agent's success in avoiding harmful outcomes, particularly critical for agents with real-world action capabilities.

5.3.4 Efficiency Metrics

Latency measures the time required for task completion, including both inference time and execution time. MAFBench [Orogat et al.(2026)] demonstrates that architectural choices can increase latency by over 100x across different frameworks.

Token Efficiency evaluates the number of tokens consumed during task completion, directly impacting operational costs.

Resource Utilization measures computational and external resource consumption, including API calls, tool invocations, and memory usage.

5.4 Evaluation Frameworks and Approaches

Different evaluation frameworks offer distinct trade-offs between automation, accuracy, and comprehensiveness.

5.4.1 Automated Evaluation

Automated evaluation leverages computational methods to assess agent performance without human intervention. LLM-as-a-Judge approaches use language models to evaluate outputs, enabling scalable assessment but potentially introducing model-specific biases.

Agent-as-a-Judge [Zhuge et al.(2024)] extends this paradigm by employing agentic systems with capabilities for intermediate feedback, process evaluation, and multi-criteria assessment. This approach achieves evaluation quality comparable to human assessment while dramatically reducing manual effort.

5.4.2 Human Evaluation

Human evaluation provides the gold standard for assessing agent behavior but requires substantial resources. Key considerations include:

- **Inter-annotator Agreement**: Ensuring consistent evaluation across multiple human judges
- **Evaluation Guidelines**: Providing clear criteria for assessing complex agent behaviors
- **Scalability**: Balancing evaluation depth with practical resource constraints

5.4.3 Environment-Based Evaluation

Environment-based evaluation assesses agents through interaction with simulated or real environments. This approach enables evaluation of emergent behaviors and unexpected failure modes that may not appear in static benchmarks.

TABLE 3: Comparison of Agent Evaluation Approaches

Approach	Automation	Accuracy	Coverage	Cost
Human Evaluation	Low	High	High	High
LLM-as-a-Judge	High	Medium	Medium	Low
Agent-as-a-Judge	High	High	High	Medium
Environment-Based	Medium	High	Medium	Medium
Adversarial	Medium	High	Low	Medium

Key challenges include environment fidelity, reproducibility of results, and the computational cost of extensive interaction trials. Frameworks like Embodied Arena [Ni et al.(2025)] address these challenges through standardized infrastructure and automated evaluation pipelines.

5.4.4 Adversarial Evaluation

Adversarial evaluation probes agent vulnerabilities through targeted attacks and stress tests. AgentGuard [Chen et al.(2025)] autonomously discovers and validates unsafe tool-use workflows, generating safety constraints to confine agent behaviors.

Nuclear Deployed [Xu et al.(2025a)] examines catastrophic risks in autonomous LLM agent decision-making, revealing that stronger reasoning abilities often increase rather than mitigate risks in high-stakes scenarios.

5.5 Comparison of Evaluation Approaches

Table 3 compares key characteristics of different evaluation approaches across multiple dimensions.

The choice of evaluation approach depends on the specific goals and constraints of the assessment. Human evaluation remains essential for validating novel capabilities and identifying subtle failure modes, while automated approaches enable scalable assessment across large task sets.

5.6 Emerging Evaluation Paradigms

Several emerging paradigms address limitations of current evaluation methods.

Continuous Evaluation recognizes that agent capabilities evolve through interaction and learning, requiring ongoing assessment rather than one-time benchmarking. This approach enables tracking of capability drift and emerging failure modes.

Compositional Evaluation assesses agents on novel combinations of known capabilities, measuring generalization beyond specific task patterns. This approach provides stronger signals about transfer learning and compositional reasoning.

Interactive Evaluation involves dynamic assessment where evaluators actively probe agent capabilities through targeted interactions, enabling discovery of edge cases and failure modes that static benchmarks miss.

5.7 Open Challenges

Despite significant progress, agent evaluation faces several open challenges:

Benchmark Obsolescence: Rapid advances in agent capabilities quickly render benchmarks obsolete, requiring continuous development of more challenging evaluation tasks.

Distribution Mismatch: Benchmarks often fail to capture the distribution of tasks encountered in real-world deployments, leading to overestimation of practical performance.

Process-Outcome Trade-offs: Optimizing for successful outcomes may incentivize behaviors that achieve results through undesirable processes, requiring careful metric design.

Safety-Capability Tension: Stronger capabilities may enable both more beneficial and more harmful behaviors, complicating the assessment of overall agent quality.

Multi-Agent Emergence: Evaluating emergent behaviors in multi-agent systems remains challenging, as system-level outcomes may not be predictable from individual agent evaluations.

These challenges motivate continued research into evaluation methodologies that can keep pace with rapidly advancing agent capabilities while providing reliable signals for safe and effective deployment.

6 Challenges and Future Directions

The rapid advancement of LLM-based autonomous agents has demonstrated remarkable capabilities across diverse domains, yet significant challenges remain before these systems can achieve reliable, safe, and general-purpose autonomy. This section examines current limitations, identifies open problems, and outlines promising research directions that could shape the future of autonomous agent systems.

6.1 Current Limitations

Despite impressive demonstrations, autonomous agents face fundamental limitations that constrain their practical deployment in real-world settings.

6.1.1 Reasoning and Planning Limitations

While LLM-based agents exhibit strong step-by-step reasoning capabilities over short horizons, they often fail to sustain coherent behavior over extended planning horizons. Recent analysis reveals a fundamental mismatch between step-wise reasoning and long-horizon planning: reasoning induces a form of step-wise greedy policy that is adequate for short horizons but fails when early actions must account for delayed consequences [Wang et al.(2026)]. This failure mode manifests as locally optimal choices that lead to early myopic commitments, systematically amplified over time and difficult to recover from.

The distinction between reasoning and planning represents a critical gap. Chain-of-thought prompting excels at decomposing immediate problems but lacks mechanisms for maintaining goal-directed behavior across extended temporal sequences. Tree-of-thought and graph-of-thought methods partially address this limitation by exploring multiple reasoning paths, yet computational costs scale poorly with planning depth, and the search process itself can diverge from task objectives.

Furthermore, agents struggle with *planning under uncertainty*. Real-world environments present stochastic dynamics, partial observability, and adversarial perturbations that

invalidate deterministic plans. Current agents lack robust mechanisms for maintaining belief states, updating plans based on unexpected observations, and gracefully degrading performance when optimal solutions become infeasible.

6.1.2 Memory and Context Constraints

Memory systems represent a critical bottleneck for agent capabilities. The fixed context window of LLMs fundamentally limits the amount of information available for reasoning, creating a tension between retaining detailed interaction history and maintaining computational efficiency.

Current memory architectures exhibit several limitations:

Retrieval-Access Trade-offs: Vector-based retrieval systems enable access to large knowledge stores but introduce retrieval errors that compound over multi-step reasoning. Irrelevant or misleading retrieved information can derail agent behavior, while critical information may be missed due to semantic mismatch between queries and stored content.

Memory Consolidation: Agents lack robust mechanisms for consolidating experiences into abstract knowledge. While humans naturally distill repeated experiences into skills and principles, current systems either store raw trajectories (expensive and inefficient) or lose critical details through aggressive summarization.

Temporal Reasoning: Understanding temporal relationships between events—causality, duration, sequencing—remains challenging. Agents often fail to maintain consistent temporal representations across long interactions, leading to contradictory beliefs and inappropriate actions.

Recent work on hierarchical memory architectures [Zhang et al.(2026)] shows promise by organizing memory into cognitively consistent episode and note structures, enabling efficient retrieval without sacrificing information fidelity. However, challenges around memory drift, conflict resolution, and self-evolution persist.

6.1.3 Tool Use and Reliability

Tool use extends agent capabilities beyond their native abilities but introduces reliability challenges. Agents must select appropriate tools from potentially large tool libraries, invoke them correctly, and handle execution failures gracefully.

Tool Selection: Retrieval-based tool selection struggles when task descriptions do not clearly map to tool capabilities. Research indicates that even strong information retrieval models exhibit poor performance on tool retrieval tasks, with low retrieval quality significantly degrading task pass rates [Shi et al.(2025)].

Error Propagation: Tool execution failures can cascade through multi-step plans. An agent that incorrectly invokes a tool may receive an error message, misinterpret the cause, and compound the error through subsequent actions. Robust error handling and recovery mechanisms remain underdeveloped.

Tool Hallucination: Agents may hallucinate tool capabilities or parameters, invoking non-existent functions or passing invalid arguments. This is particularly problematic when agents operate with unfamiliar tool libraries or when tool documentation is incomplete or ambiguous.

6.1.4 Safety and Alignment Challenges

Safety represents perhaps the most critical limitation for autonomous agent deployment. Agents with real-world action capabilities can cause harm through both commission (taking dangerous actions) and omission (failing to take protective actions).

Goal Misalignment: Agents may pursue specified objectives through unintended means, achieving task success while violating implicit constraints. Research on user-mediated attacks reveals that commercial agents bypass safety constraints in over 92% of cases when processing unverified content, converting such content into confident booking guidance [Chen et al.(2026)]. This "too helpful to be safe" phenomenon highlights the tension between helpfulness and safety.

Adversarial Vulnerabilities: Agents face diverse attack vectors including prompt injection, indirect manipulation through environment content, and long-horizon attacks that exploit multi-turn interactions. AgentLAB benchmarks demonstrate that agents remain highly susceptible to long-horizon attacks, with defenses designed for single-turn interactions failing to reliably mitigate multi-turn threats [Jiang et al.(2026)].

Emergent Misbehavior: In multi-agent systems, individually safe agents may exhibit emergent harmful behaviors through their interactions. Coordination failures, competitive dynamics, and information cascades can lead to outcomes that no individual agent would produce alone.

6.2 Open Problems

Beyond current limitations, several fundamental problems remain open for the research community.

6.2.1 Long-Horizon Autonomy

Achieving sustained autonomous operation over extended time periods presents unique challenges that current systems do not adequately address.

Temporal Consistency: Agents must maintain consistent goals, beliefs, and behaviors across interactions spanning hours, days, or longer. Current systems often exhibit drift, forgetting earlier commitments or contradicting previous statements as context accumulates.

Interruptibility and Resumption: Real-world deployment requires agents to handle interruptions gracefully and resume tasks after delays. This demands persistent state management, context reconstruction, and plan adaptation capabilities that remain underdeveloped.

Lifelong Learning: Agents operating over extended periods should improve through experience, accumulating skills and knowledge without catastrophic forgetting. The challenge lies in balancing stability (retaining existing capabilities) with plasticity (acquiring new ones) while maintaining computational efficiency.

6.2.2 Multi-Agent Coordination at Scale

While multi-agent systems have demonstrated impressive capabilities in controlled settings, scaling to larger agent populations and more complex coordination patterns remains challenging.

Communication Efficiency: As agent populations grow, communication overhead can dominate computational costs. Agents must learn when to communicate, what information to share, and with whom—balancing coordination benefits against communication costs.

Heterogeneous Coordination: Real-world deployments involve agents with diverse capabilities, objectives, and operational constraints. Coordinating heterogeneous agents requires reasoning about others' capabilities, negotiating resource allocation, and resolving conflicts—capabilities that current systems handle poorly.

Emergent Behavior Prediction: Understanding and predicting emergent behaviors in multi-agent systems remains an open problem. Small changes in individual agent policies or interaction patterns can produce qualitatively different system-level outcomes, complicating both design and verification.

6.2.3 Robust Evaluation

Current evaluation methodologies fail to capture the full complexity of agent behavior, leading to overestimation of practical capabilities.

Benchmark Saturation: Rapid capability advances quickly render benchmarks obsolete, while benchmark construction struggles to keep pace with the expanding scope of agent applications. The distribution of tasks in benchmarks often fails to reflect real-world deployment distributions.

Process-Outcome Trade-offs: Optimizing for successful outcomes may incentivize behaviors that achieve results through undesirable processes. Current metrics inadequately capture process quality, robustness, and safety alongside task success.

Reproducibility: Agent evaluation involves complex interactions with dynamic environments, making reproducibility challenging. Variations in environment state, random seeds, and non-deterministic model outputs complicate fair comparison across systems.

6.2.4 Generalization and Transfer

Agents trained or evaluated in specific environments often fail to generalize to novel settings, limiting practical applicability.

Domain Transfer: Agents developed for one domain (e.g., web navigation) often struggle when applied to related but distinct domains (e.g., mobile app navigation). Transfer learning approaches for agents remain less developed than for traditional machine learning models.

Zero-Shot Adaptation: Real-world deployment frequently requires agents to operate in environments not anticipated during development. Current systems exhibit brittle performance when facing novel tools, unexpected environment dynamics, or unfamiliar task structures.

Compositional Generalization: Agents should combine learned capabilities in novel ways to address new tasks. However, current systems often fail to exhibit compositional generalization, requiring explicit training on task combinations they will encounter.

6.3 Promising Research Directions

Addressing these challenges requires advances across multiple research fronts. We highlight several promising directions.

6.3.1 Advanced Memory Architectures

Memory system design represents a high-leverage area for improving agent capabilities.

Hierarchical Memory Structures: Inspired by human cognitive architecture, hierarchical memory systems organize information at multiple levels of abstraction—from detailed episodic traces to abstract semantic knowledge. Recent work demonstrates that such structures enable efficient retrieval while preserving information fidelity [Zhang et al.(2026)], [Lei et al.(2025)].

Continuum Memory Architectures: Moving beyond stateless retrieval, continuum memory architectures maintain and update internal state across interactions through persistent storage, selective retention, associative routing, and temporal chaining [Logan(2026)]. These architectures treat memory as a dynamic, evolving resource rather than a static lookup table.

Procedural Memory Systems: Storing and retrieving reusable procedures—skills and strategies—enables agents to compound capabilities over time. Hierarchical procedural memory with Bayesian selection and contrastive refinement has shown strong performance on multi-step tasks while enabling sample-efficient learning without LLM parameter updates [Forouzandeh et al.(2025)].

6.3.2 Improved Planning and Reasoning

Bridging the gap between reasoning and planning requires new algorithmic approaches.

Future-Aware Planning: Methods that enforce explicit lookahead, value propagation, and limited commitment can enable downstream outcomes to influence early decisions. Future-aware lookahead with reward estimation demonstrates that such approaches can allow smaller models with proper planning to outperform larger models with standard step-by-step reasoning [Wang et al.(2026)].

Hierarchical Task Decomposition: Hierarchical planning approaches that reason at multiple levels of abstraction can handle complex tasks while maintaining computational tractability. Self-organizing agent systems with tree structures demonstrate effective coordination in open-ended environments [Chen et al.(2024)].

Model-Based Planning: Integrating learned environment models into agent planning enables simulation-based reasoning about action consequences. Embodied Tree of Thoughts demonstrates how physics-based simulation can guide manipulation planning in robotic systems [Xu et al.(2025b)].

6.3.3 Safety Frameworks

Developing robust safety mechanisms is essential for trustworthy agent deployment.

Constitutional AI for Agents: Embedding explicit principles or constitutions into agent behavior provides a framework for value alignment. Multi-agent frameworks for interpretable and configurable alignment demonstrate how

constitutional principles can be enforced through agent collaboration [Masters et al.(2025)].

Verifiable Safety Constraints: Formal methods for specifying and verifying safety properties can provide guarantees about agent behavior. Research on agent safety alignment via reinforcement learning shows promise for training agents that respect specified constraints [Sha et al.(2025)].

Human-Agent Alignment: Understanding what humans want from their agents and designing for alignment across multiple dimensions—knowledge schema, autonomy, operations, reputation, ethics, and engagement—provides a foundation for trustworthy agent design [Goyal et al.(2024)].

6.3.4 Human-Agent Collaboration

Rather than fully autonomous operation, many applications benefit from effective human-agent collaboration.

Mixed-Initiative Interaction: Systems that dynamically allocate tasks between humans and agents based on capabilities, preferences, and context can leverage the complementary strengths of both. This requires agents to model human capabilities and intentions, communicate appropriately, and gracefully handle control transitions.

Explainable Agency: Agents that can explain their reasoning, decisions, and limitations enable more effective human oversight and intervention. Research on interpretable agent behavior and transparent decision-making processes supports appropriate trust calibration.

Personalization and Adaptation: Agents that adapt to individual users' preferences, workflows, and communication styles can provide more effective assistance. This requires learning from interaction history while maintaining privacy and avoiding undesirable manipulation.

6.3.5 Standardized Evaluation Infrastructure

Advancing agent capabilities requires corresponding advances in evaluation.

Process-Aware Evaluation: Evaluation frameworks that assess agent behavior throughout task execution—not just final outcomes—can identify failure modes and process deficiencies. Agent-as-a-Judge approaches demonstrate how agentic systems can evaluate other agents with intermediate feedback and multi-criteria assessment [Zhuge et al.(2024)].

Reliability Metrics: Comprehensive reliability metrics that capture consistency, robustness, predictability, and safety alongside task performance provide a more complete picture of agent capabilities [Rabanser et al.(2026)]. Such metrics decompose agent reliability along multiple dimensions, exposing limitations that aggregate success rates obscure.

Continuous Evaluation: Infrastructure for ongoing evaluation of deployed agents enables detection of capability drift, emerging failure modes, and degradation over time. This supports responsible deployment and continuous improvement.

6.4 Conclusion

The field of autonomous agents stands at an inflection point. Rapid capability advances have demonstrated the potential of LLM-based agents across diverse domains, yet fundamental challenges in reasoning, memory, safety, and evaluation constrain reliable deployment. Addressing these challenges requires coordinated progress across multiple research fronts—from algorithmic innovations in planning and memory to infrastructure advances in evaluation and safety verification.

The Brain-Perception-Action framework provides a useful lens for identifying where improvements are most needed. Current systems excel at perception and basic action execution but struggle with the brain functions that enable sustained, goal-directed autonomy. Memory systems remain limited compared to human cognitive capabilities, and multi-agent coordination lacks the robustness required for complex real-world applications.

Looking forward, the integration of advances across these dimensions—hierarchical memory, future-aware planning, constitutional safety frameworks, and human-centered design—offers a path toward more capable and trustworthy autonomous agents. The challenges are substantial, but the potential benefits of reliable autonomous systems justify sustained research investment. Success will require not only technical innovation but also careful attention to safety, alignment, and the broader societal implications of increasingly capable autonomous agents.

7 Conclusion

This survey has provided a comprehensive examination of LLM-based autonomous agents, a rapidly evolving paradigm that represents a fundamental shift in artificial intelligence research. Through the Brain-Perception-Action (BPA) framework, we have systematically organized the diverse landscape of agent architectures, capabilities, and applications, offering both conceptual clarity and practical guidance for researchers and practitioners.

7.1 Key Findings

Our analysis reveals several important insights about the current state and trajectory of autonomous agent research:

Architectural Convergence: Despite the diversity of individual systems, a coherent architectural pattern has emerged. The BPA framework—comprising central cognitive processing (Brain), environmental sensing (Perception), and action execution (Action)—provides a unified lens for understanding and comparing agent designs. This convergence reflects both the influence of cognitive science on agent design and the practical requirements of building systems that can perceive, reason, and act.

Capability Expansion: The scope of agent capabilities has expanded dramatically since the introduction of foundational frameworks such as ReAct and Chain-of-Thought prompting. Modern agents demonstrate sophisticated planning through Tree-of-Thought and Graph-of-Thought reasoning, maintain persistent memory through hierarchical architectures, coordinate effectively in multi-agent teams, and operate across diverse domains from software engineering to scientific discovery.

Application Maturity: Autonomous agents have transitioned from research curiosities to practical tools in several domains. Code development agents achieve competitive

performance on programming benchmarks, web navigation agents handle real-world websites, and multi-agent systems demonstrate effective collaboration in complex tasks. This maturation reflects both improved underlying capabilities and better understanding of domain-specific requirements.

Persistent Challenges: Despite impressive progress, fundamental challenges remain. Agents struggle with long-horizon planning, exhibit brittle generalization, face reliability issues in tool use, and present significant safety concerns. The gap between benchmark performance and real-world deployment reliability remains substantial, highlighting the need for continued research.

7.2 Contributions

This survey contributes to the field through: (1) the BPA taxonomy as a unified framework for agent analysis; (2) comprehensive coverage of over 150 papers organized by architectural components; (3) cross-domain analysis revealing common patterns and domain-specific adaptations; (4) systematic examination of evaluation methodologies and their limitations; and (5) identification of key challenges and promising research directions.

7.3 Looking Forward

The field of autonomous agents stands at a critical juncture. The convergence of increasingly capable language models, sophisticated architectural innovations, and growing practical applications suggests a trajectory toward more general and reliable autonomous systems. Yet realizing this potential requires addressing fundamental challenges in reasoning, memory, safety, and evaluation.

The most pressing research directions include: developing memory architectures that support lifelong learning without catastrophic forgetting; bridging the gap between step-by-step reasoning and long-horizon planning; establishing robust safety frameworks that prevent harmful behaviors while maintaining helpfulness; and creating evaluation methodologies that capture the full complexity of agent behavior.

Ultimately, the success of autonomous agents will be measured not only by their capabilities but by their trustworthiness, reliability, and alignment with human values. The frameworks, taxonomies, and insights presented in this survey aim to accelerate progress toward this vision while ensuring that advances serve beneficial ends. As agents become increasingly integrated into human workflows and decision-making processes, the responsibility for thoughtful, safety-conscious development has never been greater.

Acknowledgment

The authors thank the anonymous reviewers for their valuable feedback. This work was supported in part by [funding sources to be added].

References

[1] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed. Pearson, 2020.

[2] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan *et al.*, "Language models are few-shot learners," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.

[3] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang *et al.*, "A survey on large language model based autonomous agents," *arXiv preprint arXiv:2308.11432*, 2023.

[4] T. R. Sumers, S. Yao, K. Narasimhan, and T. L. Griffiths, "Cognitive architectures for language agents," *Transactions on Machine Learning Research*, 2024, arXiv:2309.02427.

[5] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 24 824–24 837.

[6] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," in *Advances in Neural Information Processing Systems*, vol. 36, 2023.

[7] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, and T. Scialom, "Toolformer: Language models can teach themselves to use tools," *arXiv preprint arXiv:2302.04761*, 2023.

[8] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing reasoning and acting in language models," in *International Conference on Learning Representations*, 2023.

[9] X. Ma, Y. Gao, Y. Wang, R. Wang, X. Wang *et al.*, "Safety at scale: A comprehensive survey of large model and agent safety," *arXiv preprint arXiv:2502.05206*, 2025.

[10] Y. Liu, W. Chen, Y. Bai, X. Liang, G. Li *et al.*, "Aligning cyber space with physical world: A comprehensive survey on embodied AI," *arXiv preprint arXiv:2407.06886*, 2024.

[11] J. E. Laird, A. Newell, and P. S. Rosenbloom, "SOAR: An architecture for general intelligence," *Artificial Intelligence*, vol. 33, no. 1, pp. 1–64, 1987.

[12] J. R. Anderson and C. Lebiere, "ACT-R: A simple theory of complex cognition," *American Psychologist*, vol. 51, no. 4, pp. 355–365, 1996.

[13] A. S. Rao and M. P. Georgeff, "BDI agents: From theory to practice," pp. 312–319, 1995.

[14] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[15] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 421–436, 2018.

[16] S. V. Albrecht and P. Stone, "Autonomous agents and multi-agent systems," *Foundations and Trends in Machine Learning*, vol. 11, no. 3-4, pp. 243–437, 2018.

[17] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess *et al.*, "Scaling laws for neural language models," *arXiv preprint arXiv:2001.08361*, 2020.

[18] L. Wang *et al.*, "A survey on large language model based autonomous agents," *arXiv preprint arXiv:2308.11432*, 2023.

[19] J. Chen, Y. Jiang, J. Lu, and L. Zhang, "S-Agents: Self-organizing agents in open-ended environments," 2024.

[20] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *arXiv preprint arXiv:2201.11903*, 2022.

[21] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," *arXiv preprint arXiv:2305.10601*, 2023.

[22] Y. Yao, Z. Li, and H. Zhao, "Beyond chain-of-thought, effective graph-of-thought reasoning in language models," *arXiv preprint arXiv:2305.16582*, 2023.

[23] L. Yang, Z. Yu, T. Zhang, S. Cao, M. Xu *et al.*, "Buffer of thoughts: Thought-augmented reasoning with large language models," *Advances in Neural Information Processing Systems*, 2024, arXiv:2406.04271.

[24] N. Shinn, F. Cassano, E. Berman, A. Gopinath, K. Narasimhan, and S. Yao, "Reflexion: Language agents with verbal reinforcement learning," in *Advances in Neural Information Processing Systems*, vol. 36, 2023.

[25] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, L. Gao *et al.*, "Self-refine: Iterative refinement with self-feedback," in *Advances in Neural Information Processing Systems*, vol. 36, 2023.

[26] R. Wang, J. Zhang *et al.*, "Learning from failure: Integrating negative examples when fine-tuning large language models as agents," *arXiv preprint arXiv:2402.11651*, 2024.

[27] C. Packer, S. Wooders, K. Lin, V. Fang, S. G. Patil *et al.*, "MemGPT: Towards LLMs as operating systems," *arXiv preprint arXiv:2310.08560*, 2023.

[28] Y. P. Shkolnikov, "Agent memory below the prompt: Persistent Q4 KV cache," *arXiv preprint arXiv:2603.04428*, 2026.

[29] J. S. Park, J. C. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein, "Generative agents: Interactive simulacra of human behavior," *arXiv preprint arXiv:2304.03442*, 2023.

[30] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao *et al.*, "Voyager: An open-ended embodied agent with large language models," *arXiv preprint arXiv:2305.16291*, 2023.

[31] Q. Xie *et al.*, "Embodied-RAG: General non-parametric embodied memory," *arXiv preprint arXiv:2409.18313*, 2024.

[32] H. He, W. Yao, K. Ma, W. Yu, Y. Dai *et al.*, "WebVoyager: Building an end-to-end web agent with large multimodal models," in *Annual Meeting of the Association for Computational Linguistics*, 2024.

[33] Y. Pan, D. Kong, S. Zhou, C. Cui, Y. Leng *et al.*, "WebCanvas: Benchmarking web agents in online environments," *arXiv preprint arXiv:2406.12373*, 2024.

[34] H. Wen *et al.*, "AutoDroid: LLM-powered task automation in Android," in *ACM International Conference on Mobile Computing and Networking*, 2023.

[35] Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang, "HuggingGPT: Solving AI tasks with ChatGPT and its friends in Hugging Face," *arXiv preprint arXiv:2303.17580*, 2023.

[36] C. Li, J. Liang, A. Zeng, X. Chen, K. Hausman *et al.*, "Chain of code: Reasoning with a language model-augmented code emulator," in *International Conference on Machine Learning*, 2024, arXiv:2312.04474.

[37] W. Xu *et al.*, "Embodied tree of thoughts: Deliberate manipulation planning," *arXiv preprint arXiv:2512.08188*, 2025.

[38] S. Hong, M. Zhuge, J. Chen, X. Zheng, Y. Cheng *et al.*, "MetaGPT: Meta programming for a multi-agent collaborative framework," in *International Conference on Learning Representations*, 2024, arXiv:2308.00352.

[39] Y. Yu, Z. Yao, H. Li, Z. Deng, Y. Cao *et al.*, "FinCon: A synthesized LLM multi-agent system with conceptual verbal reinforcement for enhanced financial decision making," *arXiv preprint arXiv:2407.06567*, 2024.

[40] D. Yang, J. Wei, M. Li, J. Liu, L. Liu *et al.*, "MedAide: Information fusion and anatomy of medical intents via LLM-based agent collaboration," *arXiv preprint arXiv:2410.12532*, 2024.

[41] Z. Luo, C. Xu, P. Zhao, Q. Sun, X. Xi, C. Zhang, J. Zheng, S. Fan, X. Li, and Y. Huang, "WizardCoder: Empowering code large language models with Evol-Instruct," in *International Conference on Learning Representations*, 2023.

[42] W. Shen, C. Li, H. Chen, M. Yan, X. Quan *et al.*, "Small LLMs are weak tool learners: A multi-LLM agent," *arXiv preprint arXiv:2401.07324*, 2024.

[43] M. Haseeb, "Context engineering for multi-agent LLM code assistants using Elicit, NotebookLM, ChatGPT, and Claude code," *arXiv preprint arXiv:2508.08322*, 2025.

[44] F. V. Ruiz, "Agent-driven automatic software improvement," in *International Conference on Evaluation and Assessment in Software Engineering*, 2024.

[45] R. Hoda, "Toward agentic software engineering beyond code," in *International Conference on Software Engineering*, 2025.

[46] K. Yang *et al.*, "AgentOccam: A simple yet strong baseline for LLM-based web agents," *arXiv preprint arXiv:2410.13825*, 2024.

[47] Y. Guo, C. Guo, A. Sun, H. He, X. Yang *et al.*, "Web-CogReasoner: Towards knowledge-induced cognitive reasoning for web agents," *arXiv preprint arXiv:2508.01858*, 2025.

[48] F. Chen, T. Wu, V. Nguyen, and C. Rudolph, "Too helpful to be safe: User-mediated attacks on planning and web-use agents," *arXiv preprint arXiv:2601.10758*, 2026.

[49] S. Liu, Y. Li, K. Zhang, Z. Cui, W. Fang *et al.*, "Odyssey: Empowering Minecraft agents with open-world skills," *arXiv preprint arXiv:2407.15325*, 2024.

[50] S. Mao *et al.*, "ALYMPICS: LLM agents meet game theory," *arXiv preprint arXiv:2311.03220*, 2023.

[51] C. Park *et al.*, "Do LLM agents have regret? a case study in online learning and games," in *International Conference on Learning Representations*, 2024.

[52] F. Bredell, H. A. Engelbrecht, and J. C. Schoeman, "Augmenting the action space with conventions to improve multi-agent cooperation in Hanabi," *Autonomous Agents and Multi-Agent Systems*, vol. 39, no. 28, 2025.

[53] J. Huang *et al.*, "An embodied generalist agent in 3D world," in *International Conference on Machine Learning*, 2023.

[54] M. Zawalski *et al.*, "Robotic control via embodied chain-of-thought reasoning," *arXiv preprint arXiv:2407.08693*, 2024.

[55] S. Shaji, F. Huppertz, A. Mitrevski, and S. Houben, "From language to action: Can LLM-based agents be used for embodied robot cognition?" in *IEEE International Conference on Robotics and Automation*, 2026.

[56] J. Mao *et al.*, "A language agent for autonomous driving," in *Conference on Language Modeling*, 2023.

[57] H. Zhang *et al.*, "BadRobot: Jailbreaking embodied LLMs in the physical world," in *International Conference on Learning Representations*, 2024.

[58] P. Trirat *et al.*, "AutoML-Agent: A multi-agent LLM framework for full-pipeline AutoML," in *International Conference on Machine Learning*, 2024.

[59] S. Rothfarb *et al.*, "Hierarchical multi-agent large language model reasoning for autonomous functional materials discovery," *arXiv preprint arXiv:2512.13930*, 2025.

[60] M. Xiao *et al.*, "Knowledge-driven agentic scientific corpus distillation framework," *arXiv preprint arXiv:2504.19565*, 2025.

[61] P. Shojaee *et al.*, "LLM-SRBench: A new benchmark for scientific equation discovery," in *International Conference on Machine Learning*, 2025.

[62] Y. Xiao, E. Sun, D. Luo, and W. Wang, "TradingAgents: Multi-agents LLM financial trading framework," *arXiv preprint arXiv:2412.20138*, 2024.

[63] G. Li, R. Wu, and H. Tan, "A plan reuse mechanism for LLM-driven agent," *arXiv preprint arXiv:2512.21309*, 2024.

[64] T. Zhang, X. Zhang, R. Cools, and A. L. Simeone, "Focus agent: LLM-powered virtual focus group," in *ACM International Conference on Intelligent Virtual Agents*, 2024.

[65] M. Zhuge *et al.*, "Agent-as-a-judge: Evaluate agents with agents," *arXiv preprint arXiv:2410.10934*, 2024.

[66] A. Orogat *et al.*, "Understanding multi-agent LLM frameworks: A unified benchmark," *arXiv preprint arXiv:2602.03128*, 2026.

[67] A. Zharmagambetov *et al.*, "AgentDAM: Privacy preservation in web-navigation agents," *arXiv preprint arXiv:2502.04596*, 2025.

[68] Z. Ying *et al.*, "SecureWebArena: A holistic security evaluation benchmark for vision-language model-based web agents," *arXiv preprint arXiv:2501.07880*, 2025.

[69] C. Gao *et al.*, "EmbodiedCity: A benchmark platform for embodied agent," *arXiv preprint arXiv:2410.09604*, 2024.

[70] F. Ni *et al.*, "Embodied arena: A comprehensive evaluation platform," *arXiv preprint arXiv:2509.15273*, 2025.

[71] Z. Guo *et al.*, "BEDI: A systematic benchmark for UAV-embodied agents," *arXiv preprint arXiv:2503.11234*, 2025.

[72] J. Light, M. Cai, S. Shen, and Z. Hu, "AvalonBench: Evaluating LLMs playing the game of Avalon," *arXiv preprint arXiv:2310.05036*, 2023.

[73] S. Park *et al.*, "Orak: A foundational benchmark for LLM gaming agents," *arXiv preprint arXiv:2503.08765*, 2025.

[74] Y. Guo *et al.*, "PPTC benchmark: Evaluating large language models for PowerPoint task completion," *arXiv preprint arXiv:2311.01767*, 2023.

[75] Z. Zhang *et al.*, "PPTC-R benchmark: Towards evaluating the robustness," *arXiv preprint arXiv:2403.03788*, 2024.

[76] X. Wang, S. Zhang, W. Zhang, W. Dong, J. Chen *et al.*, "ZSC-Eval: An evaluation toolkit for multi-agent zero-shot coordination," in *Advances in Neural Information Processing Systems Datasets Track*, 2023.

[77] Z. Shi *et al.*, "Retrieval models aren't tool-savvy: Benchmarking tool retrieval for large language models," in *Proceedings of the Association for Computational Linguistics*, 2025.

[78] Z. Zhang, S. Cui, Y. Lu, J. Zhou, J. Yang *et al.*, "Agent-safetybench: Evaluating the safety of LLM agents," *arXiv preprint arXiv:2412.14470*, 2024.

[79] X. Yuan *et al.*, "R-Judge: Benchmarking risk awareness for LLM agents," *arXiv preprint arXiv:2401.10319*, 2024.

[80] Y. Chen *et al.*, "AgentGuard: Active safety constraints for LLM agents," *arXiv preprint arXiv:2501.02362*, 2025.

[81] R. Xu *et al.*, "Nuclear deployed: Analyzing catastrophic risks in decision-making of autonomous LLM agents," *arXiv preprint arXiv:2502.11355*, 2025.
[82] Z. Wang, F. Wu, H. Wang, X. Tang, B. Li *et al.*, "Why reasoning fails to plan: A planning-centric analysis of long-horizon decision making in LLM agents," *arXiv preprint arXiv:2601.22311*, 2026.
[83] N. Zhang, X. Yang, Z. Tan, W. Deng, and W. Wang, "HiMem: Hierarchical long-term memory for LLM long-horizon agents," *arXiv preprint arXiv:2601.06377*, 2026.
[84] T. Jiang, Y. Wang, J. Liang, and T. Wang, "AgentLAB: Benchmarking LLM agents against long-horizon attacks," *arXiv preprint arXiv:2602.16901*, 2026.
[85] M. Lei, Y. Zhao, G. Wang, Z. Mai, S. Cui *et al.*, "STMA: A spatio-temporal memory agent for long-horizon embodied task planning," *arXiv preprint arXiv:2502.10177*, 2025.
[86] J. Logan, "Continuum memory architectures for long-horizon LLM agents," *arXiv preprint arXiv:2601.09913*, 2026.
[87] S. Forouzandeh, W. Peng, P. Moradi, X. Yu, and M. Jalili, "Learning hierarchical procedural memory for LLM agents through Bayesian selection and contrastive refinement," *arXiv preprint arXiv:2512.18950*, 2025.
[88] C. Masters *et al.*, "ARCANE: A multi-agent framework for interpretable and configurable alignment," in *AAAI Workshop*, 2025.
[89] Z. Sha *et al.*, "Agent safety alignment via reinforcement learning," *arXiv preprint arXiv:2507.08270*, 2025.
[90] N. Goyal, M. Chang, and M. Terry, "Designing for human-agent alignment: Understanding what humans want from their agents," *arXiv preprint arXiv:2404.04289*, 2024.
[91] S. Rabanser, S. Kapoor, P. Kirgis, K. Liu, S. Utpala *et al.*, "Towards a science of AI agent reliability," *arXiv preprint arXiv:2602.16666*, 2026.