Exploring the Development Status of Agents Based on Large Language Models

Zixi Chen

{chenzixi26offer@163.com}

College of International Tourism and Public Administration, Hainan University, Haikou City, Hainan Province, 570228, China

Abstract. The blistering development of Large Language Models (LLMs) is leading to a paradigm shift of artificial intelligence. Of these advancements, agents constructed on the creation of LLM are transforming the models of human-computer interaction and productivity. The given paper is an attempt to analyze the technology of the agents based on the LLM in a systematic way and build a comprehensive framework of cognitions based on micro-level technical systems, up to macro-level patterns of collaboration. It summarizes that the neutralization of the constraints of conventional software can be achieved by the use of the LLM-based agents, which include a set of the brain, planning, memory and the ability to utilize the tools in an unstructured environment. Enterprise-level restructuring, embod intelligence and Agentic AI will be such directions as the development of LLM-based agents in the future.

Keywords: LLMs; Agents; Multi-Agent Systems; Embodied Intelligence; Agentic AI

1 Introduction

The development of the newest Large Language Models (LLMs) in recent years to include GPT-5.1, Gemini 3.0, and Grok 4.1 has made the representational and generalization potentials of artificial intelligence soar to unprecedented levels. Nonetheless, with these strong reasoning capacities, the LLMs are fundamentally a passive brain which has no mechanism of any interaction with the external physical or virtual world, therefore it is hard to have them complete complex tasks that need long-term planning letting the brain act independently. In order to succeed this weakness and give LLMs hands and feet and memory, LLM-based agents have been developed quickly becoming one of the focal points of academic research and implementation in industries.

A LLM-based agent is an artificial intelligence system or agent, which does not require human intervention. It is a combination of the intelligence and the capacity of LLMs with the implementation capabilities of external tools that allow it to analyze various problems using information about the environment and contexts, make rational decisions about them, and perform various tasks to reach particular objectives by itself.

Although the research on the topic of LLM-based agents has exploded, one can observe a

ICIAAI 2026, May 22-24, Singapore, Singapore

DOI 10.4108/eai.22-5-2026.2365080

shortage of systematic integration of the available literature on surveys. The existing revisions are mainly scattered reviews of the internal technical aspects of the LLM-based agents or individual studies of the collaboration mechanisms of several agents, without a single view that views the micro-level technical structures as the organic component of the macro-level pathways. Particularly, the long-term memory of the conversation aspect as investigated by Maharana et al. in the evaluation study offers a detailed discussion of the technical implementation of the memory module on its own but does not consider the synergies among memory and other technical modules (planning and tool usage) in long-term memory approaches[1]. On the other hand, the survey by Li et al. on multi-agent systems (MASs) on a more systematic analysis of the workflows and infrastructure of multi-agent collaboration based on LLM provides an analysis only concerning one of the modes of operation, single agent and hybrid agent without providing a systematic classification and comparative analysis of the different modes of operation of a multi-agent system[2]. The theses of this paper are intended to bridge this gap by developing a cumulative review framework, named Technical Architecture -> Operational Modes -> Future Evolution, to present a comprehensive overview coverage of all the evolutional agents technologies built on the basis of LLM, thus able to give a panoramic view of the entire evolutionary process of individual cognitions to collective intelligences, and to a higher extent, to physical interaction.

This paper seeks to review systematically agent technologies based on LLM according to the above context. The agents, as described in this paper, have better natural language understanding and complex task decomposition capabilities as compared to the traditional agents. The paper dissects the basic technical architecture of agents, the main four components of which include the brain, the planning, the memory, and the tool use. It then offers a comparative study of their functioning and working systems in three aspects; single agent system, multi-agent system, and hybrid agent system. Lastly, it explores the existing technical and ethical issues, and provides a prospectus to the further development of enterprise-level agents, embodiment intelligence, and Agentic AI, in the aim of making the future development of agents and their application an available source of reference.

2 Agent Technical Architecture

Agents have become the most basic change of state in the paradigm of AI: changing into active problem solvers, capable of autonomous planning and execution, rather than passive content generators responding to instructions. In contrast to conventional AI systems, agents do not just have strong natural language processing capacities, but as a result of combination of planning, memory and tool-use modules, they eventually represent a complex computational system with respect of the sense of the environment and act independently. Based on the currently accepted mainstream paradigms of research, the overall structure of an agent is captured by the following formula:

$$\text{Agent} = \text{LLM} + \text{Planning} + \text{Memory} + \text{Tool Use} \quad (1)$$

The architecture goes beyond the restrictions of conventional software based on fixed regulations. Their generic cognitive abilities and reasoning support enable it to manifest

greater strength and flexibility when faced with open-ended and complicated unstructured tasks than ever before.

2.1 LLM

The LLM plays an important role in the agent architecture, as it is the main controller and the brain of the agent. It does not only perform natural language understanding and generation but also performs central functions of storage of information, logical reasoning, understanding instructions, and decision making. The maximum capacity of the model clearly defines the performance criteria of the agent in a complicated situation. There is a current trend in research, which reveals that foundational LLMs are evolving beyond content generators and into situation builders, with effective reasoning skills. It means that models do not only have to respond to questions, but they should possess deep semantics and be able to provide the way of further actions.

In deciding on whether or not a foundational LLM could be the brain of an agent, it is the coding performance and tool-use performance that must be analyzed. These two elements are not disconnected, code can be used as the rationale underlying in which an agent applies complicated tool use. The natural language ambiguity is not sufficient to handle the requirements of control with fine precision when it is required of the data analysis or enterprise level applications. To create logical operations and data extraction, an agent can create structured Python or SQL, which may be used to cleverly make use of external tools, thereby providing the basis on which an agent may carry out its autonomous action system. At the same time, such a high-speed performance feature cannot work without a robust long-context understanding since complicated chains of the tasks are usually supported by enormous quantities of background knowledge. This reasoning is strongly justified by the study by Lu et al. in the auditing domain- by designing a collaborative architecture to combine a long-text analysis expert (i.e., Moonshot) and a logical reasoning expert (i.e., DeepSeek-R1) the system scored better when used to analyze complex documents and structured information than single models. This also shows that an ideal agent brain must be an organic integration of very specific performance ability and wide understanding capacity[3].

2.2 Planning

The planning module gives an agent the ability to think to cope with complicated tasks, which is the most important aspect that allows agents to develop out of simple reactive to deliberative systems. An agent cannot depend on intuition to know what to do when faced with high level goals which are large, unclear, or abstract; it needs to realize the capacity to break down higher level goals into a sequence of executable logically rigorous sequences of subgoals. This is an operation which combines not only the foresight of actions in advance, but also its assessment of the present and in order to arrange a favorable course to reach the objective.

In order to improve the reasoning of the LLMs on challenging problems, based on the analysis of complex tasks, researchers have suggested some reasoning paradigms. Chain of Thought (CoT) technique is the most basic one. It leads the model to the show of intermediate reasoning steps and thus partitions a complicated logical issue into a lineal process of thinking which enhances the performance of the model on mathematical and logical assignments

significantly. Linear reasoning falls short however of dealing with open-ended problems, which involve the exploration of several possibilities. In order to overcome this drawback, Tree of thoughts (ToT) was established. ToT enables the model to come up with a variety of the possible branches at each step in reason, forming a tree of thoughts. It will then run algorithms like depth first search or breadth first search to examine, look forward and backtrack on various thought processes, which allows a search of a globally optimal solution in the solution space.

At the concrete implementation level, the Reasoning + Acting (ReAct) model has turned into a universal option in the sphere of the industry. ReAct is a clever framework which entails integrating explicit reasoning with action execution whereby the agent is bound to produce reasoning traces on the situation and thereafter perform the action, and be able to observe the consequence. This is a dynamic adjustment process of Observe-Think-Act whereby future plans are changed by this cyclic mechanism according to the feedback of the real time environment which consequently decreases the likelihood of hallucinations occurring. Moreover, to consider making the system even stronger, Shinn et al. came up with the Reflexion mechanism. It is a mechanism that adds a step in self-critique: when a task does not succeed or result is not satisfactory, the agent casts a look back at the past paths, through the linguistic feedback mechanism and evaluates the cause of the failure and makes the experience become part of the long-term memory. This will enable the agent to prevent the repetition of errors in the future work, self-improvement[4].

2.3 Memory

Memory module tackles the constraints posed by statelessness of LLMs making agents have the ability to learn past interactions and experiences with the course of time, and, thus, remain consistent in their behavior. In agent systems, memory is generally divided into sensory memory, short-term memory and long-term memory that make up the knowledge retention mechanism of the agent.

Sensory memory is the gateway of information processing of an agent which is the temporary buffering of environment raw signals. In multimodal agents, this is represented by the initial feature integration and storage of visual images, audio streams or real-time text streams. In their survey Li et al. observe that despite the fact that sensory memory has a tremendously brief span of existence, it is upon which agent knowledge of complex environments is based. Transmission of only important information that goes through the attention mechanism to the short term memory is then processed in the memory further. This algorithmically removes the potential of overloading limited computational power with huge noise polluting the environment[2].

Working memory, which is also known as short term memory, is directly equivalent to the context window of the LLM. It is in charge of storing information pertaining to current work, most recent conversation history, and intermediate states in the course of reasoning. The Transformer architecture has a limitation on the size of its context window because of the computational complexity, and so short-term memory has a fixed size, which can only store so much data. To make an effective use of such limited resource, methods of context compression, summarization, or sliding windows are normally used so that no critical information should be forgotten.

By the introduction of external storage media, long-term memory tackles the problem of context window limitations and gives the agent the benefit of a virtually unlimited reserve of knowledge. Technically, the implementation of long-term memory mostly relies on the use of the vector databases in the construction. The agent encodings the information or experience collected to a set of vectors to be stored. The agent uses Retrieval-Augmented Generation (RAG) technology when solving a new problem by choosing semantic similarity to quickly access and recollect previous experience or external knowledge and inject it into the context at hand. Yet, studies by Maharana et al. show that, although simple relying on retrieval augmentation may bring the factual information, there are yet limitations in the accommodation of the long dependence temporal and cause reasoning, which imply that the future memory systems should be developed based on a simple retrieval method than the more intensive structured understanding[1]. In addition, it is required to store snapshots of the world states in the memory module to assist the agent in backtracking and tracking the state when performing complex actions where it is highly dependent on how the implicit state changes through the multiple turns of the interaction between the agent and the environment.

2.4 Tool Use

The basic aspect which can be identified by which agents stand out of traditional conversational chatbots is the use of tools, which is used to mark the shifting of AI, who is now an actor rather than a speaker. An agent can invoke external APIs, run software systems and even manage physical hardware by equipping an executor, and this way may greatly widen its functionality. An example is that an agent could utilize a calculator to perform specific mathematical tasks, call on a search engine to get real-time news, a code interpreter to analyze a large amount of data, or even directly connect to a CRM or ERP system provided by a business, to support business processes.

There is an interface between LLM and external tools at the level of technical implementation which is the function calling mechanism. In this framework, developers should give the LLM the description of tools in the prompt in the form of the JSON Schema, the name of the function, the description of the functionalities, and the structure of the parameters. Upon a user giving an instruction, the LLM semantically analyzes and reason to find out whether a tool call is required. Should that be the case, the model does not produce the normal natural language output but a structured JSON object with the name of the particular function and parameters. When this JSON is received, an external system will run the logic behind that particular functionality and respond (for example, with data in an API response) to the model in text form. Combining this implementation outcome, the model eventually generates a human understandable response or moves to the subsequent phase. Yet, as was found in the ToolSandbox benchmark by Lu et al. under the set of complex toolchains with implicit state dependencies (e.g. one must log in before a tool may be invoked), even the current models are also prone to making missteps in the invocation sequence. This implies that the tool-use ability does not solely depend on single-purpose recognition but further promotes state tracing and causal process[5]. To standardize this process and make it simpler, the industry has been designing a set of standard interface protocols and middleware, like API Bank and MCP. The infrastructures create tool interfaces and environment description that can be called upon by agents in a more convenient manner, creating an expanded range of third-party services, and establishing the foundation of a large-scale, interrelated ecosystem of agent applications.

3 Classification of Agent Architectures

The problem paths and limitation of an agent are defined by the working modes. Depending on the count of agents, interaction structures, and human or environmental involvement, the currently existing agents have mainly developed into three types which include single agent, multi-agent, and hybrid agent.

3.1 Single Agent

The single agent is the minimal functional unit that independently completes the entire process from perception and planning to execution. Its operational mechanism typically follows an autonomous cycle of "Thinking-Planning-Executing," driven by a single LLM. The core advantages lie in its simple architecture, high execution efficiency, and the absence of complex communication overhead. A typical single agent, such as AutoGPT, iteratively solves problems through continuous self-prompting. However, Tran et al. critically point out in their survey that single agents are severely constrained by the limited context window and reasoning capabilities of a single model. When handling long-chain tasks, they are highly prone to falling into pitfalls such as "hallucination accumulation" or "dead loops." Due to the lack of correction mechanisms from external perspectives, single agents struggle to address complex problems requiring cross-domain knowledge integration. Consequently, they are more suitable for automated tasks with clear objectives and short chains, but exhibit significant limitations when dealing with unstructured, complex long-horizon problems in open-world scenarios[6].

3.2 Multi-Agent

To break the boundaries of a solitary model MASs are emulated by a group of human collaborative phenomena via a large number of agents doing all sorts of different tasks (product manager, architect, and engineer) to resolve complicated issues. The study by Li et al. shows that MASs attain emergence of collective intelligence, that surpasses the historical addition of the constituents, with a divide-and-conquer approach, which greatly enhances the success of complex tasks[2].

Adding further, Tran et al. further categorize the collaboration mechanisms in MASs into three different categories that are, cooperative, competitive, and co-opetitive as shown in Fig. 1[6]. In collaborative architectures such as MetaGPT, the agents execute their assigned functions, using standard operating procedures, making the chaos likely to occur during tasks execution low. Competitive architectures like LLMArena or multi-agent debate systems, in contrast, use confrontation and questioning between agents to encourage more in-depth reasoning, which is a reasonable solution to the problems of blind conformity and hallucinations. The intelligence of co-opetition architectures is that both approaches are intelligently combined to provide the best of each. Considering an example, where there is a code generation task, there is first off competition amongst multiple agents to develop different solutions, and then a cooperative mechanism comes in and refines the merits of each proposal to create the final code. This strategy guarantees variance of the solution space as well as improvement of the aggregate output quality. As much as MASs are good at managing complexity, it has its fair share of challenges. Since more agents the more agents increases the Redundancy of communication and cost of coordination exponentially increases. As a result, one of the important research

challenges in the field has been to come up with effective communication topologies, e.g. striking a balance between decentralization and hierarchical structures.

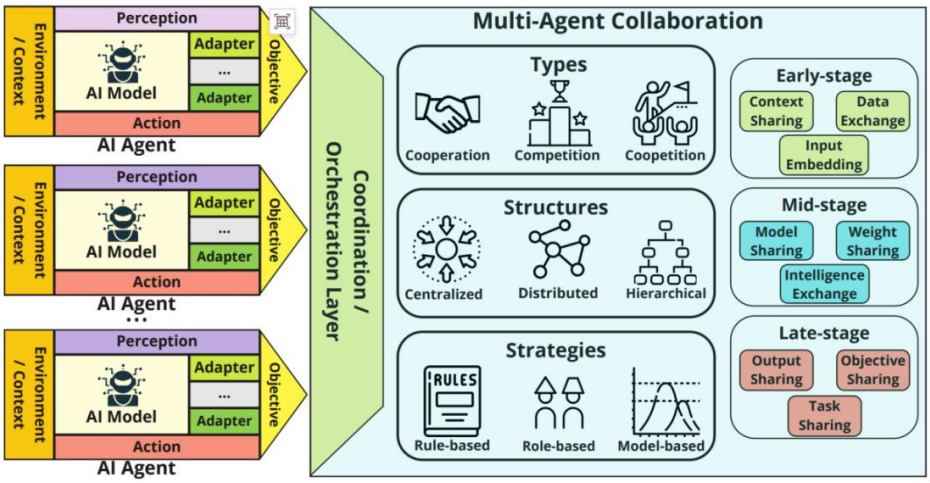

Fig. 1. Collaboration Mechanisms in MASs[6].

3.3 Hybrid Agent

Hybrid agents seek to resolve the shortcomings of pure AI systems both in robustness and alignment, and usually it requires collaboration between humans and machines or with the environment. A Sensor-in-the-loop architecture suggested by Ren et al. incorporates sensor data of the physical world (e.g. heart rate, location) into the decision making loop of the agent. This will allow the agent to not only comprehend the text, but also get to read the physiology of the user and even the environmental response, hence offer more customized and reliable services[7]. In such a way adding human feedback or physical validation to the purely digital agents enables them to overcome the weakness of the latter with regard to both during-deployment properties, i.e., real-world deployment, and with regard to the ethical properties, i.e., ethical alignment, it is a major step towards high-reliability applications like healthcare and autonomous driving.

The three categories of agents are the agents with the different driving cores, therefore each possesses their own advantages and constraints and perform better in unique application cases in a synopsis as noted in Table 1. In practice, architecture selection is not in mutually exclusive terms but a trade off based on the complexity of the tasks, response time and system fault-tolerance. The developers must find an ideal balance between the efficiency of single agent execution, reasoning of MASs and the reliability of the deployment of the hybrid agents depending on the requirements of the scenario.

Table 1. Comparison of Single Agent, Multi-Agent, and Hybrid Agent

Comparison Dimension	Single Agent	Multi-Agent	Hybrid Agent
Driving Core	Single LLM	Multiple LLMs with role	LLM +

Comparison Dimension	Single Agent	Multi-Agent	Hybrid Agent
		division	Human/Sensors
Advantages	Simple architecture, fast response (<1s), low deployment cost	Collective intelligence, strong task decomposition, good fault tolerance	High robustness, environmental perception, human-machine collaboration, ethical control
Disadvantages	Prone to hallucinations, gets lost in long-horizon tasks, limited generalization	High communication overhead (latency 2-5s), complex coordination logic, potentially reduced efficiency	High system complexity, reliant on external resources, high cost
Typical Applications	Intelligent customer service, document summarization, simple Q&A	Complex project development (MetaGPT), multi-step research & analysis (AutoGPT), code generation (ChatDev)	Autonomous driving, remote medical diagnosis, industrial quality inspection, disaster rescue
Scalability	Limited (vertical scaling)	Strong (horizontal scaling)	Medium (dependent on external resources)
Suitable Scenarios	Well-defined, structured tasks, high real-time response requirements	Tasks requiring multi-domain knowledge, decomposable complex tasks, high fault-tolerance requirements	High-uncertainty environments, tasks requiring real-time perception or human intervention, high safety requirements

4 Challenges and Prospects

Although significant breakthroughs have been done by the agents in terms of representational and generalization capability, there are still deeper issues that the field of agents confronts in terms of transitioning between the experimental environments and large-scale industrial application, such as system robustness, long-range planning in a way that is irregular, and safety and ethical considerations. The agent technology is now under a critical capability re-shaping period. Its evolution has transcended the tool empowerment frontline and moved into the direction of the establishment of trusted interaction contracts to entrench security foundations and rooted deep into enterprise core systems to repenstruct business processes. Moreover, there is a physical shift of the agents towards an "embodied" state instead of a disembodied algorithmic state, and eventually even full autonomy states of the Agentic AI, which will be a fundamental shift of the auxiliary digital assistants to a universal productivity forces.

4.1 Technical and Ethical Challenges

The main issue of the existing agent technology is reliability and security of the system. The fact that LLMs provide generalization skills to agents is that the very problem of them being a hallucination is multiplied exponentially on the action level: a problem with text generation can result in misinformation but any mistake with action planning can create disastrous cascading failures. Empirical study by Ji et al. states that under long chain tasks agents tend to produce oftentimes plausible and practically unsolvable plans more than others because they cannot properly recognize limitations imposed by environmental factors[8]. Moreover, since the agents are provided with long-term memory, the protection against the leakage of sensitive information during the interaction process and mitigation of malicious commands injections become important issues that need urgent addressing. The architecture of future should have a firmer sandboxes execution system and human in the loop supervision pattern so that although the agents have their way, they do not overstep the boundaries that humans have set.

4.2 Enterprise-Level Agents

Enterprise-level agents at the application level are being slowly transformed into no longer being mere simple knowledge-based assistants, instead into deep business process rearchitects. Although modern Copilot modes are mainly concerned with efficiency gains, the future Agent modes will eliminate the productivity issues on their level. These are not going to merely be viewed as agents that process unstructured text, but will become highly integrated into a fundamental part of business like ERP and CRM applications, and will assume the roles of resource allocation and making decisions independently. This transformation requires that agents have inbuilt capabilities of managing heterogeneous data and working in conformity to the savings of complex enterprise rules. As examples, in software engineering, the cooperation of agents with expert knowledge has facilitated the end-to-end automation of the multi-stage lifecycle processes of requirements analysis through code delivery[9]. It is an indication that the agents are increasingly becoming more than just software tools and transforming into a digital employee, which is the key driving force of enterprise digital transformation.

4.3 Agent Empowermen

Agent architectures are essential in the creation of embodied intelligence by enabling critical foundational support through the creation of closed loop perception, decision-making, and action. Embodied intelligence marks a critical boundary point in the history of agents, as it is the shift of the in the digital domain to bodied agents that have the ability to engage in physical interactions. The traditional agents may also be limited by their disembodied character and ability to do only symbolic processing in the virtual spaces. Rather, it is the essence of embodied intelligence to furnish the agent with his physical body by furnishing his brain. This physical jump allows intelligent agents to be able to go beyond passively processing information. They attain real-time interaction and action in the physical world, through the closed loop of perception and cognition, decision-making, and action. Embodied intelligence provides agents with human like capabilities of self directed learning as well as environmental adaptability by the incorporation of intelligent systems as part of attempts to create physical agents like robots or autonomous vehicles, embedded into their design. This is one of the fundamental changes of the mere thinkers into the actors, which can perceive and change the physical world.

4.4 The Ultimate Form of Agent Development

The development of agent technology is leading to the development of Agentic AI intelligent systems with complete autonomy. The agents of the future will not be servants who comply with the order but will be co-workers who are able to offer insights into needs, goals, courses of action, and will learn in the context of dynamism[10]. This will be when the weaknesses of individual intelligence will be overcome. By extensive cooperative interaction, competition and evolution between multiple agents, systems have a chance to produce an apparent collective intelligence that is larger than the aggregate of individual intelligences. Such intelligence may be able to handle global-scale complicated issues. Nevertheless, it also exerts more pressure on the interpretability of the models and value correspondence. The ultimate research question will be to ensure that the self-driven development of agents is always aimed at the basic interests of humanity.

5 Conclusion

The current advancements in the development of the LLM-based agent technologies are systematically reviewed in this paper. We find that with the four essential ingredients used, namely the brain (chosen), planning, and memory, as well as, tool utilization, the capabilities of the LLM-based agents have made an exemplary breakthrough. Planning module allows the decomposition of tasks and logical thinking and memory module facilitates constant storage and management of state information thus unraveling the restraints of traditional models of state that were constant. In addition, the tool-use mechanism enables agents to perform extrinsic actions which are beyond their semantic comprehension, but covers their practical action.

Applying operations mode, it is possible to classify the based agents of LLM as follows: single agent has great expertise in efficiency but is limited on capacity; MASs apply communication to emergent collective intelligence, optimum in complex tasks but at a higher coordination cost; hybrid agents adopt human or environmental feedback meaning that they show unique value in high-reliability contexts.

In the future, the examples of LLM based agent technology include aspects that are moving into more fundamentally integrated enterprise level applications, embodied intelligence and interaction with environments, and highly autonomous Agentic AI. Nonetheless, its development is still experiencing immense setbacks in terms of reliability, safety and ethical alignment. Providing such evolution to meet the basic human interests is the ultimate requirement of future studies.

References

[1] Maharana, A., et al.: Evaluating very long-term conversational memory of LLM agents. *arXiv Preprint*. pp. arXiv:2402.17753 (2024)

[2] Li, X.Y., et al.: A survey on LLM-based multi-agent systems: Workflow, infrastructure, and challenges. *Vicinagearth*. pp. 9 (2024)

[3] Lu, Y., Hao, J.X., Tang, X.S.: Dual-model synergy for audit opinion prediction: A collaborative LLM agent framework approach. *International Review of Economics & Finance*. pp. 104642 (2025)
[4] Shinn, N., et al.: Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*. pp. 8634–8652 (2023)
[5] Lu, J.R., et al.: Toolsandbox: A stateful, conversational, interactive evaluation benchmark for LLM tool use capabilities. *Findings of the Association for Computational Linguistics: NAACL 2025*. pp. 1–?? (2025)
[6] Tran, K.T., et al.: Multi-agent collaboration mechanisms: A survey of LLMs. *arXiv Preprint*. pp. arXiv:2501.06322 (2025)
[7] Ren, Z.W., et al.: Toward sensor-in-the-loop LLM agent: Benchmarks and implications. *Proceedings of the 23rd ACM Conference on Embedded Networked Sensor Systems*. pp. 1–?? (2025)
[8] Ji, Z.L., et al.: Testing and understanding erroneous planning in LLM agents through synthesized user inputs. *arXiv Preprint*. pp. arXiv:2404.17833 (2024)
[9] He, J.D., Treude, C., Lo, D.: LLM-based multi-agent systems for software engineering: Literature review, vision, and the road ahead. *ACM Transactions on Software Engineering and Methodology*. pp. 1–30 (2025)
[10] Bandi, A., et al.: The rise of agentic AI: A review of definitions, frameworks, architectures, applications, evaluation metrics, and challenges. *Future Internet*. pp. 404 (2025)