# Agent Trajectory Explorer: Visualizing and Providing Feedback on Agent Trajectories

**Michael Desmond, Ja Young Lee, Ibrahim Ibrahim,**
**James M. Johnson, Avirup Sil, Justin MacNair, Ruchir Puri**

IBM Research

**Abstract**

Agentic systems interleave large language model (LLM) reasoning, tool usage, and tool observations over multiple iterations to tackle complex tasks. The raw data from an agent's problem-solving process (the agents' trajectory) is not an ideal format for human analysis and oversight. There is a need for tooling that converts this primary data into an easily navigable and understandable visual format for better human feedback. To address this opportunity, we developed the *Agent Trajectory Explorer*, a tool designed to help AI developers and researchers visualize, annotate, and demonstrate agent behavior.

**Video** — https://youtu.be/tosCdzT4MFE

## Introduction

Current agentic systems, such as ReAct (Yao et al. 2022) and SWE-Agent (Yang et al. 2024), sequentially interleave powerful LLM reasoning, tool calling, and tool observations to solve complex tasks. Specifically, an LLM generates a thought which may invoke the use of a tool from an available catalog, the tool is executed by the framework, and the output is appended to the LLM context as an observation. The next thought/action is then conditioned on all prior accumulated context. Using this pattern an agent can work through problems that would not be possible using an LLM alone.

The raw representation of an agents' problem solving behavior, or trajectory (a potentially large context prompt incorporating instructions, reasoning, tool calling and outputs), is generally not ideal for human analysis. More importantly, arriving at an answer or solution via a potentially long sequence of LLM-driven reasoning and interaction steps brings the issue of human oversight to the forefront. There is a need for tooling to convert raw agent trajectory information into a visual representation that is easy to navigate and understand by humans, and ideally provides a medium for expression of feedback.

To address this need, we developed the *Agent Trajectory Explorer*[1], a tool designed to help agent developers and researchers to examine, annotate, and demonstrate agent behavior in a convenient manner.



[1]We plan to open source the tool upon internal approval.

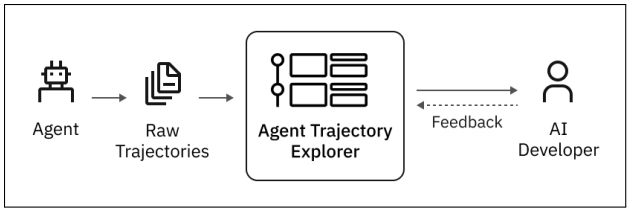


Figure 1: The Agent Trajectory Explorer consumes raw agent trajectories representing agent behavior. Trajectories are then formatted and visually presented. The system also provides an opportunity to capture feedback.

## Agent Trajectory Explorer

The *Agent Trajectory Explorer* is designed to enable AI agent developers and researchers to examine agent trajectories, and provide feedback for future improvement. The system is a web based application implemented using the Next.js web development framework (Vercel 2024). See figure 1 for a general overview of the application and how it is positioned in the agentic systems workflow.

### Formatting

The *Agent Trajectory Explorer* supports a formatting plug-in framework enabling users to convert their own trajectory files into a standard representation consumable by the tool.

At a system level, the user configures the application with a directory containing a set of agent trajectories (in any valid json format), and an appropriate formatting plug-in (a typescript class (TypeScript 2024)) that is responsible for converting raw json files to framework constructs that are then used for exploration and visualization features.

Multiple trajectory formats can co-exist within the tool by specifying a mapping that matches a top level type field in the raw json to a formatting plugin. At runtime the system looks for the type field and chooses the corresponding formatter. This can be useful when working with multiple agentic frameworks.

### The Visual Paradigm

By default the *Agent Trajectory Explorer* assumes a Thought/Action/Observation (TAO) visual paradigm inspired by ReAct (Yao et al. 2022). Agent behavior is ex-

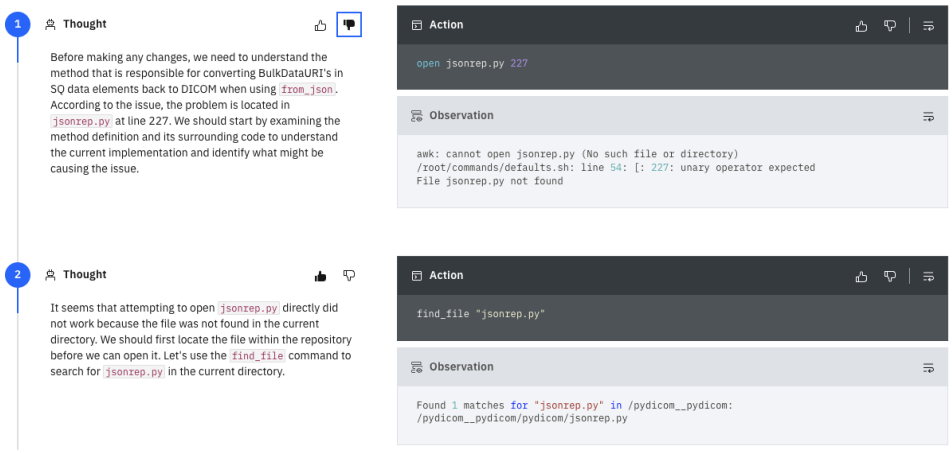


Figure 2: The trajectory is displayed as a sequence of turns, each of which contains a thought, action and observation.

pressed as a linear sequence of TAO triples. TAO is easy to understand, conveniently ordering the agents problem solving process into reasoning, tool calling, and tool outputs. See figure 2. Each of the TAO elements are rendered in a unique way to facilitate ease of differentiation.

While a TAO paradigm covers many current agentic use-cases we acknowledge that other paradigms exist (Xu et al. 2023; Shinn et al. 2024) and compatibility with the TAO visual model may vary. Its also unlikely that a universal visual paradigm will suffice given the flexibility of composition when applying the agentic pattern. As such, we expect that the trajectory explorer will evolve to support new and existing visual paradigms over time.

The explorer also supports a raw presentation mode wherein the agent behavior is displayed as a sequence of raw context elements (similar to the SWE-Agent inspector (Yang et al. 2024)), reflecting the LLM inferencing process that occurs as the agent is executed to generate new turns. As such the tool provides multiple levels of abstraction, catering to different types of users, one focusing on the agents reasoning and actions, and the other on the raw inference.

## Feedback

While the primary function of the Trajectory Explorer is presentation of agent behavior, the tool does support a feedback mechanism. Users can indicate positive or negative feedback on particular thoughts and actions within a trajectory (see fig 2). While this feedback modality is limited, we expect to add a richer set of feedback controls to the tool over time, such as the ability to rewrite thoughts and actions, and addition of commentary that can provide a basis for implementing behavioral improvements.

## Discussion

The Trajectory Explorer Tool is useful when working with agents that solve complex problems over a series of turns using an observational reasoning pattern, and developers need to review the agents' behavior and provide feedback. The tool also fills a niche of presenting the agent to other stakeholders due to the easy to understand abstractions, and the presentation that focuses on core behavioral elements.

However, we expect that agent trajectories will only get more complex as new LLM interaction patterns are discovered (Xu et al. 2023), and more complicated domains are exposed to the technology (Jimenez et al. 2023). Alternative patterns of reasoning and interaction beyond linear TAO such as trees (Yao et al. 2024) and graphs are more challenging constructs to both visualize and understand. And more recently multi-agent systems (Guo et al. 2024; Xu et al. 2023; Wang et al. 2024) focus on agent profiles, interactions between agents, and collective decision-making processes, which introduce an additional layer of complexity for humans to understand. As agents become more complex, the need for supportive tools like the Agent Trajectory Explorer Tool will become essential.

Beyond the current scope of the Trajectory Explorer Tool, we believe that interactive development and debugging of agentic systems is important. In the context of chat bots Constitution Maker (Petridis et al. 2024) demonstrated the value of interactive critique. Similar functionality, in the context of agentic systems, would help users design and implement agents in a more natural iterative manner. Developing effective methods for providing human feedback, and understanding how to apply this feedback to the agent will be crucial. Similarly, features like comparing trajectories before and after changes, or between different versions of an agent, will help AI developers work more effectively.

## Conclusion

In conclusion, compound AI systems like agents can now solve complex problems through a combination of LLM reasoning, tool usage, and environmental observations. As a result, AI developers need effective methods for reviewing these behaviors to debug issues, observe patterns, and provide feedback. To address this need, we present the *Agent Trajectory Explorer*, a tool designed to help agent developers analyze agent trajectories and offer feedback for continuous improvement.

## References


Guo, T.; Chen, X.; Wang, Y.; Chang, R.; Pei, S.; Chawla, N. V.; Wiest, O.; and Zhang, X. 2024. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*.

Jimenez, C. E.; Yang, J.; Wettig, A.; Yao, S.; Pei, K.; Press, O.; and Narasimhan, K. 2023. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*.

Petridis, S.; Wedin, B. D.; Wexler, J.; Pushkarna, M.; Donsbach, A.; Goyal, N.; Cai, C. J.; and Terry, M. 2024. Constitutionmaker: Interactively critiquing large language models by converting feedback into principles. In *Proceedings of the 29th International Conference on Intelligent User Interfaces*, 853–868.

Shinn, N.; Cassano, F.; Gopinath, A.; Narasimhan, K.; and Yao, S. 2024. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36.

TypeScript. 2024. TypeScript: JavaScript With Syntax For Types.

Vercel. 2024. Next.js by Vercel - The React Framework.

Wang, X.; Li, B.; Song, Y.; Xu, F. F.; Tang, X.; Zhuge, M.; Pan, J.; Song, Y.; Li, B.; Singh, J.; et al. 2024. OpenDevin: An Open Platform for AI Software Developers as Generalist Agents. *arXiv preprint arXiv:2407.16741*.

Xu, B.; Peng, Z.; Lei, B.; Mukherjee, S.; Liu, Y.; and Xu, D. 2023. Rewoo: Decoupling reasoning from observations for efficient augmented language models. *arXiv preprint arXiv:2305.18323*.

Yang, J.; Jimenez, C. E.; Wettig, A.; Lieret, K.; Yao, S.; Narasimhan, K.; and Press, O. 2024. Swe-agent: Agent-computer interfaces enable automated software engineering. *arXiv preprint arXiv:2405.15793*.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T.; Cao, Y.; and Narasimhan, K. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.