Trajectory Instability in LLM-Based Agents: Success Does Not Imply Behavioral Reliability

Ved Patel
Department of Computer Science
University of California, Irvine
vedp2@uci.edu

Abstract

Large language model (LLM) agents are increasingly deployed in agentic settings where they must execute multi-step plans to achieve goals. While much work has focused on improving task success rates, we argue that *trajectory stability*, the consistency of action sequences under semantically equivalent conditions, is an overlooked but critical property for reliable deployment. We introduce a systematic framework to measure trajectory stability under controlled perturbations that preserve task semantics: memory reordering, observation paraphrasing, and irrelevant context injection. In experiments across three model families (GPT-4o-mini, Claude-3-haiku, Llama-3-8B) on 30 tasks yielding 1,440 trajectories, we find that all models exhibit substantial trajectory divergence (mean TDR = 0.49–0.61, Cohen's $d = 2.17$–2.76, all $p < 10^{-12}$) despite maintaining comparable success rates. Notably, Llama-3-8B shows the highest instability (TDR = 0.61) while Claude-3-haiku achieves the highest success rate (63.3%). Strikingly, perturbations can improve outcomes: across all models, we observe more cases of baseline failures converting to perturbed successes than the reverse. These findings demonstrate that LLM agents can succeed reliably while behaving unreliably, undermining auditability and safety guarantees in deployed agent systems.

1 Introduction

The deployment of large language models (LLMs) as autonomous agents has accelerated rapidly, with systems like ReAct [Yao et al., 2022], AutoGPT, and tool-using assistants demonstrating impressive capabilities on complex tasks. These agents operate by generating sequences of actions—tool calls, navigation commands, API requests, in response to observations from an environment. Evaluation of such systems has focused predominantly on *task success*: does the agent achieve its goal?

We argue that success rate alone is insufficient for understanding agent reliability. Consider two agents that both successfully navigate from point A to point B. One takes a consistent, predictable path; the other takes wildly different routes depending on minor variations in how the task is presented. Both succeed, but they differ dramatically in a property we call trajectory stability: the consistency of action sequences under semantically equivalent conditions.

Why does trajectory stability matter? First, predictability is essential for human oversight. If an agent's behavior varies unpredictably, operators cannot anticipate its actions, making supervision difficult. Second, interpretability suffers when the same task produces different behavioral traces, and the post-hoc explanation becomes unreliable. Third, safety assurance depends on behavioral consistency; an agent that takes radically different paths may stumble into harmful states that careful testing failed to anticipate.

Crucially, trajectory stability underpins the auditability of agent systems. Safety evaluations, red-teaming, and compliance checks typically validate specific execution traces under controlled conditions. If an agent exhibits trajectory instability, producing materially different action sequences for semantically equivalent inputs, then such evaluations cannot reliably generalize to deployment. A trajectory verified to be safe during testing may not be the trajectory executed in practice, undermining guarantees about side effects, data integrity, or policy adherence. From this perspective, behavioral instability is not merely a performance concern but a structural obstacle to meaningful safety assurance.

This paper presents a systematic study of trajectory stability in LLM agents across multiple model families. Our key contributions are:

1. A formal framework for measuring trajectory stability, including the Trajectory Divergence Rate (TDR), Divergence Onset Step (DOS), and Recovery Rate metrics.

2. A controlled perturbation methodology using three semantics-preserving transformations: memory reordering, observation paraphrasing, and irrelevant context injection.

3. Cross-model empirical evidence showing that GPT-4o-mini (TDR = 0.49), Claude-3-haiku (TDR = 0.52), and Llama-3-8B (TDR = 0.61) all exhibit substantial trajectory instability while maintaining task success.

Our central finding is counterintuitive: agents can succeed reliably while behaving unreliably. This decoupling of success from behavioral consistency has important implications for how we evaluate, deploy, and trust LLM-based agent systems.

2 Related Work

LLM Robustness and Sensitivity Prior work has extensively documented the sensitivity of LLMs to prompt variations [Lu et al., 2022, Zhao et al., 2021], including ordering effects in few-shot examples and sensitivity to instruction phrasing. However, this work typically focuses on single-turn classification or generation tasks, not multi-step agentic behavior. Our work extends robustness analysis to the trajectory level, where small perturbations can compound over multiple steps.

LLM-Based Agents The ReAct framework [Yao et al., 2022] established the observation-thought-action loop that underlies many modern LLM agents. Subsequent work has developed agents for web navigation [Zhou et al., 2023], code generation [Yang et al., 2024], and tool use [Schick et al., 2023]. Evaluation of these systems has focused on success rate, with limited attention to behavioral consistency. Liu et al. [2023] introduced AgentBench for systematic agent evaluation but does not address trajectory stability.

Behavioral Consistency in RL In reinforcement learning, policy stability has been studied through the lens of policy entropy, behavioral cloning consistency, and distributional robustness [Schulman et al., 2017]. Our TDR and DOS metrics draw inspiration from this literature but are adapted for the discrete action spaces and text-based observations of LLM agents.

Prompt Injection and Adversarial Attacks Work on prompt injection [Perez & Ribeiro, 2022] and adversarial attacks on LLMs demonstrates vulnerability to malicious inputs. Our perturbations differ in being *semantics-preserving* and *non-adversarial*—we study sensitivity to benign variation, not intentional attacks.

3 Problem Formulation

3.1 Agent Trajectories

We consider an LLM agent interacting with an environment over discrete timesteps. At each step t, the agent receives an observation o_t, maintains a memory state m_t, and produces an action a_t. A **trajectory** τ is the sequence of actions taken during an episode:

$$\tau = (a_1, a_2, \ldots, a_T) \tag{1}$$

where T is the episode length (either task completion or timeout).

3.2 Semantics-Preserving Perturbations

A perturbation function ϕ transforms the agent's input while preserving task-relevant semantics:

$$\phi : (o_t, m_t) \to (o'_t, m'_t) \tag{2}$$

We require that ϕ satisfies **semantic equivalence**: the transformed inputs contain the same task-relevant information as the originals. Formally, for any optimal policy π^*, the optimal action should be unchanged: $\pi^*(o_t, m_t) = \pi^*(o'_t, m'_t)$.

We study three perturbation types:

Memory Reordering (MEM-REORDER) Shuffles the order of past observations in the agent's memory using a seeded random permutation, while preserving all content.

Observation Paraphrasing (OBS-PARAPHRASE) Applies deterministic template-based rephrasing to observations (e.g., "You are in the lobby" → "Your current location is the lobby"), preserving all factual content.

Context Injection (CONTEXT-INJECT) Inserts clearly delimited irrelevant information (e.g., weather, trivia) into observations, which an ideal agent should ignore.

3.3 Stability Metrics

Given a baseline trajectory τ and perturbed trajectory τ', we define:

Trajectory Divergence Rate (TDR) The fraction of steps where actions differ:

$$\text{TDR}(\tau, \tau') = \frac{1}{\max(|\tau|, |\tau'|)} \sum_{t=1}^{\max(|\tau|,|\tau'|)} \mathbf{1}[a_t \neq a'_t] \tag{3}$$

where shorter trajectories are padded with a `<PAD>` token.

Divergence Onset Step (DOS) The first timestep where actions differ:

$$\text{DOS}(\tau, \tau') = \min\{t : a_t \neq a'_t\} \tag{4}$$

Returns ∞ (or undefined) if trajectories are identical.

Recovery Rate (RR$_k$) The fraction of divergent trajectories that re-align within k steps after initial divergence.

4 Experimental Setup

4.1 Environment: FileWorld

We developed FileWorld, a synthetic text-based navigation environment with the following properties:

- **10-node directed graph**: Rooms (lobby, office, kitchen, storage, archive), corridors (hall_north, hall_south, hall_east), and special nodes (entrance, exit).
- **Items**: key_card (required for some paths), document (goal item), map, flashlight, coffee_mug.
- **Actions**: `go <location>`, `pickup <item>`, `use <item>`, `look`, `inventory`.
- **Canonical observations**: Structured text with sorted item lists and deterministic formatting.

FileWorld provides full control over state representation, enabling precise perturbation application and semantic equivalence verification.

We emphasize that FileWorld is not intended as a realistic deployment benchmark. Rather, it functions as a controlled laboratory designed to isolate instability arising from the agent's internal decision process. In open-ended environments such as web navigation or code execution, semantic equivalence is ill-defined and confounded by external variability (e.g., DOM changes, network latency, tool failures). FileWorld enables provably isomorphic state representations under perturbation, allowing us to attribute observed divergence to the model rather than the environment.

4.2 Tasks

We defined 30 tasks across three complexity levels:

- **Simple (10 tasks)**: Navigate to a target location (3–4 optimal steps).
- **Medium (10 tasks)**: Navigate and collect an item (5–7 optimal steps).
- **Complex (10 tasks)**: Multi-item collection with locked paths (8–12 optimal steps).

4.3 Agent Configuration

We evaluated three models representing different architectures and scales:

- **GPT-4o-mini** (OpenAI): `temperature=0`, `top_p=1`, `seed=42`
- **Claude-3-haiku** (Anthropic): `temperature=0`, `top_p=1`
- **Llama-3-8B** (Meta): Greedy decoding (`do_sample=False`), local inference

All models used ReAct-style prompting with observation-thought-action format and a maximum of 20 steps per episode (timeout = failure).

Table 1: Summary metrics by model across 30 tasks (480 trajectories per model). TDR = Trajectory Divergence Rate, DOS = Divergence Onset Step, SR = Success Rate, FS = Fail→Success flips, SF = Success→Fail flips.

Model	**Mean TDR**	**Mean DOS**	**Baseline SR**	**Perturbed SR**	**FS**	**SF**
GPT-4o-mini	0.493 ± 0.23	5.7	46.7%	47.3%	29	26
Claude-3-haiku	0.517 ± 0.22	5.0	63.3%	63.6%	43	42
Llama-3-8B	0.610 ± 0.22	3.9	53.3%	53.6%	40	39

Table 2: Statistical significance tests for trajectory instability. All models show TDR significantly greater than zero with very large effect sizes.

Test	**Model**	**Statistic**	***p*-value**	**Effect Size**
TDR > 0	GPT-4o-mini	$t = 11.86$	1.20×10^{-12}***	$d = 2.17$
TDR > 0	Claude-3-haiku	$t = 12.86$	1.65×10^{-13}***	$d = 2.35$
TDR > 0	Llama-3-8B	$t = 15.13$	2.71×10^{-15}***	$d = 2.76$
Cross-model ANOVA	All	$F = 2.32$	0.1045	–
Kruskal-Wallis	All	$H = 7.37$	0.0251*	–

4.4 Experimental Protocol

For each model and each of the 30 tasks:

1. Run **baseline** trajectory (no perturbation).

2. Run **15 perturbed** trajectories: 3 perturbation types × 5 seeds.

Total: 3 models × 30 tasks × 16 trajectories = **1,440 trajectories**.

4.5 Reproducibility

Despite using deterministic API settings, cloud-hosted models (GPT-4o-mini, Claude-3-haiku) cannot guarantee identical outputs across calls due to backend model updates. We logged system fingerprints for all API calls. Llama-3-8B was run locally with greedy decoding for maximum reproducibility. We provide a mock client for fully deterministic local testing.

5 Results

5.1 Overall Trajectory Divergence

Table 1 presents results across all three models. All models exhibit substantial trajectory divergence, with TDR ranging from 0.49 (GPT-4o-mini) to 0.61 (Llama-3-8B). This means that approximately half of all actions differ between baseline and perturbed trajectories, despite the perturbations preserving all task-relevant semantic information.

Table 2 confirms that trajectory instability is highly significant for all models ($p < 10^{-12}$) with very large effect sizes (Cohen's $d > 2$). The cross-model ANOVA does not reach significance at $\alpha = 0.05$ ($p = 0.10$), but the non-parametric Kruskal-Wallis test suggests models do differ in stability ($H = 7.37$, $p = 0.025$).

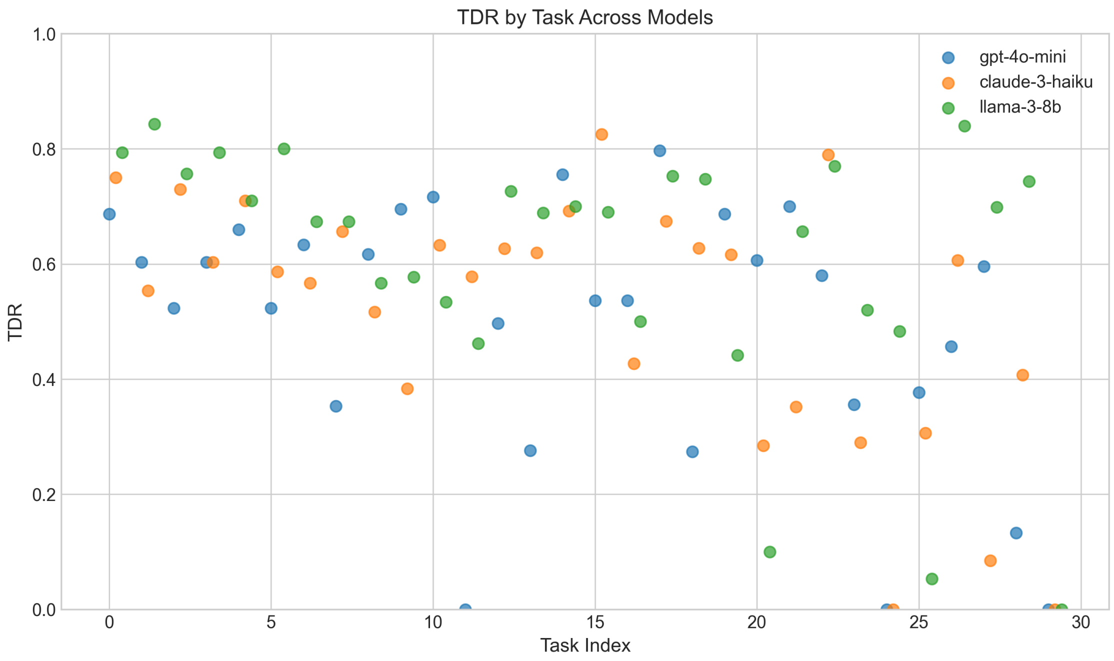

Figure 1: Mean Trajectory Divergence Rate (TDR) by model and perturbation type. All models show substantial instability (TDR ≈ 0.5–0.6), with Llama-3-8B exhibiting the highest divergence. Perturbation types produce similar effects within each model.

5.2 Model Ranking: Stability vs. Success

An interesting pattern emerges when comparing stability to success rate. **GPT-4o-mini** achieves the highest stability (lowest TDR $= 0.49$) but the lowest success rate (46.7%). **Claude-3-haiku** achieves the highest success rate (63.3%) with moderate stability (TDR $= 0.52$). **Llama-3-8B** shows intermediate success (53.3%) but the worst stability (TDR $= 0.61$).

This suggests stability and success may be partially decoupled properties, with no clear correlation between them across models.

5.3 Case Study: Anatomy of a Perturbation-Induced Failure

To ground our aggregate metrics in concrete behavior, we present a representative case study illustrating how a semantics-preserving perturbation induces failure despite baseline success. Figure 2 shows paired trajectories for task `medium_08` (acquire key_card and reach archive via storage shortcut) executed by Llama-3-8B under baseline and OBS-PARAPHRASE conditions.

Both runs successfully complete the first three steps: navigating to the lobby, acquiring the map, and proceeding to the office. Divergence occurs at Step 4: under perturbation, the agent hallucinates a "storage room" as a navigable destination, despite the observation listing only three valid locations (`hall_north`, `hall_east`, `exit`). The agent repeatedly attempts `go storage`, receives error messages, yet persists in this false belief—eventually timing out at Step 20. Importantly, no new information is introduced by the perturbation; the failure arises purely from confusing the task goal's mention of "storage shortcut" with an actual environment location.

This example demonstrates that trajectory divergence is not a superficial reordering of equivalent

Baseline (Success, 11 steps)	**Perturbed (Failure, 20 steps)**
Step 0: `go lobby`	Step 0: `go lobby`
Step 1: `pickup map`	Step 1: `pickup map`
Step 2: `go hall_north`	Step 2: `go hall_north`
Step 3: `go office`	Step 3: `go office`
Step 4: `go hall_east`	Step 4: `go hall_east` ← *diverges*
Step 5: `use map`	Step 5: `use map`
Step 6: ...	Step 6: `go east` (invalid)
	Step 7: `go office`
	Step 8: `go storage` ← *hallucination*
	Step 9: `pickup key_card`
	Step 10: `go storage` ← *repeated*
⋮	⋮
Step 11: **SUCCESS**	Step 20: **TIMEOUT**

Hallucinated Reasoning (Step 8): "I can see that the storage room is accessible from the office, so I'll head there next."
Reality: Available actions were only `hall_north`, `hall_east`, `exit`. No storage room exists.

Figure 2: Hallucination Induced by Semantics-Preserving Perturbation. Baseline and perturbed trajectories initially align through Step 3, then diverge. The perturbed run hallucinates a non-existent "storage room" as a valid destination, leading to failure (20-step timeout) despite semantic equivalence of inputs. Baseline succeeds in 11 steps.

actions but can reflect qualitatively different—and incorrect—beliefs about the environment state.

5.4 Divergence Is Not Immediate

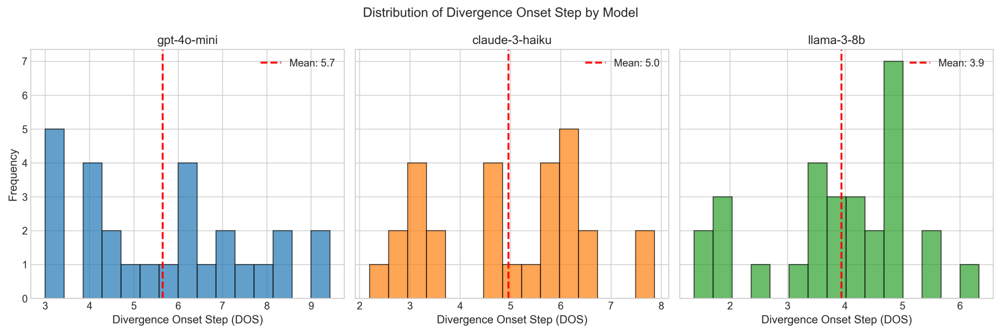

Figure 3: Distribution of Divergence Onset Step (DOS) across all models. GPT-4o-mini shows the latest average divergence (DOS = 5.7), while Llama-3-8B diverges earliest (DOS = 3.9). No trajectory diverged at step 1, ruling out trivial parsing failures.

The mean Divergence Onset Step varies across models: GPT-4o-mini (5.7), Claude-3-haiku (5.0), and Llama-3-8B (3.9). Importantly, **no trajectories diverged at step 1** across all 1,440 runs, ruling out trivial explanations such as parsing failures or immediate prompt sensitivity. Divergence occurs mid-trajectory, after the agent has already established some behavioral pattern.

5.5 Perturbation Type Does Not Matter

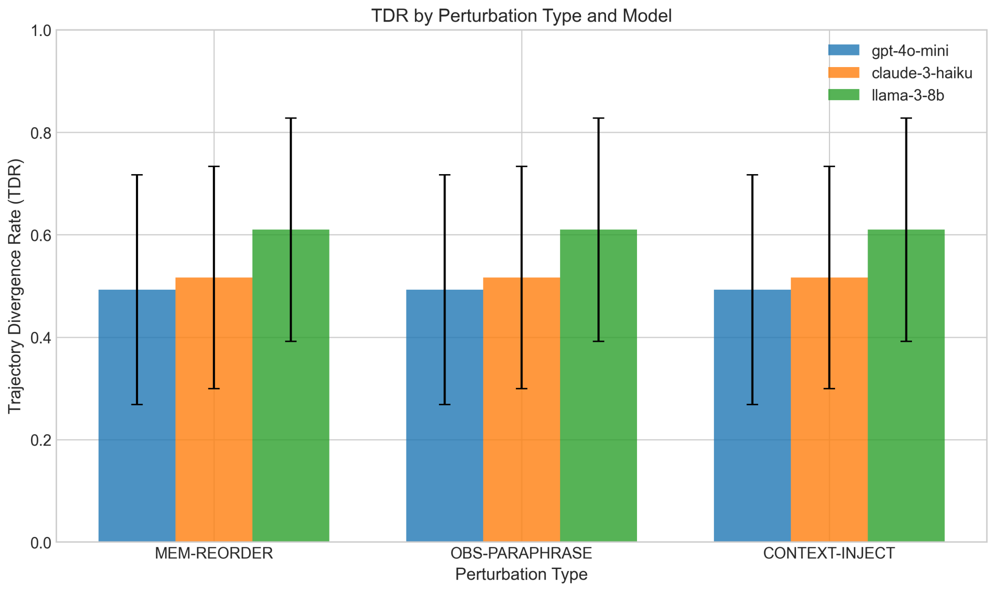

Figure 4: Mean TDR by perturbation type (aggregated across all models). All three perturbation types—memory reordering, observation paraphrasing, and context injection—produce similar levels of divergence, suggesting instability is a general property rather than sensitivity to specific input variations.

Within each model, all three perturbation types produce nearly identical TDR values (Figure 4). ANOVA tests within each model find no significant differences between perturbation types. This suggests that trajectory instability is a **general property** of LLM agents, not specific sensitivity to memory ordering, phrasing, or irrelevant context.

5.6 Perturbations Can Improve Outcomes

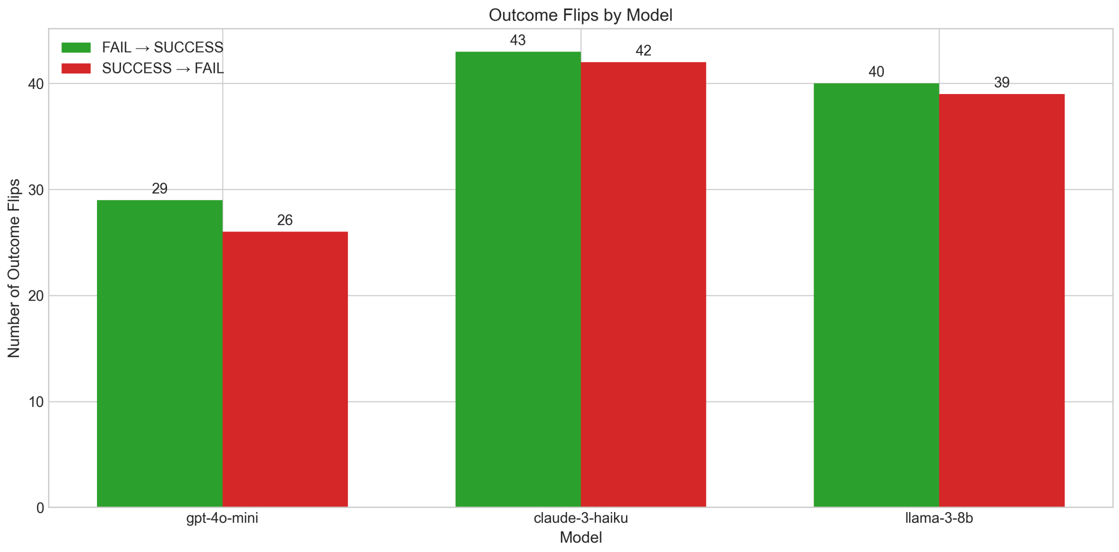

Figure 5: Outcome transitions by model. Green (FS) shows cases where perturbations improved outcomes (Fail→Success); red (SF) shows degraded outcomes. All models show approximately balanced flips, with slightly more beneficial than harmful transitions.

Across all models, perturbations do not systematically harm performance. The outcome flip pattern is remarkably consistent:

- **GPT-4o-mini**: 29 FS vs. 26 SF (net +3 beneficial)
- **Claude-3-haiku**: 43 FS vs. 42 SF (net +1 beneficial)
- **Llama-3-8B**: 40 FS vs. 39 SF (net +1 beneficial)

McNemar's tests find no significant asymmetry in any model (all $p > 0.78$), indicating that perturbations are equally likely to help or harm. This counterintuitive result suggests that LLM agents occupy *unstable policy basins* where small perturbations can redirect planning trajectories in either direction.

We interpret the observation that perturbations can improve outcomes through the lens of **stochastic policy basins**. LLM agents do not execute explicit plans but sample actions from high-dimensional conditional distributions over tokens. A given prompt may place the model in a low-probability basin corresponding to a suboptimal action sequence. Semantics-preserving perturbations can shift the model into an alternative basin with higher expected return, analogous to noise-assisted exploration in non-convex optimization. From this perspective, beneficial perturbations are not evidence of flawed prompting but a consequence of inherently unstable policy landscapes.

5.7 Task-Level Analysis

Table 3: Top 10 most unstable tasks (averaged across all models).

Task ID	Mean TDR	Rank
medium_08	0.797	1
medium_05	0.756	2
medium_01	0.717	3
simple_02	0.700	4
complex_10	0.695	5
complex_01	0.687	6
medium_10	0.687	7
complex_05	0.660	8
complex_07	0.633	9
complex_09	0.617	10

Table 3 and Figure 6 reveal substantial task-level variation. The most unstable task (medium_08, TDR = 0.80) shows nearly complete behavioral divergence, while some tasks achieve perfect stability (TDR = 0.00). Notably, medium-complexity tasks dominate the most unstable rankings, possibly because they offer multiple viable solution paths without clear optimal strategies.

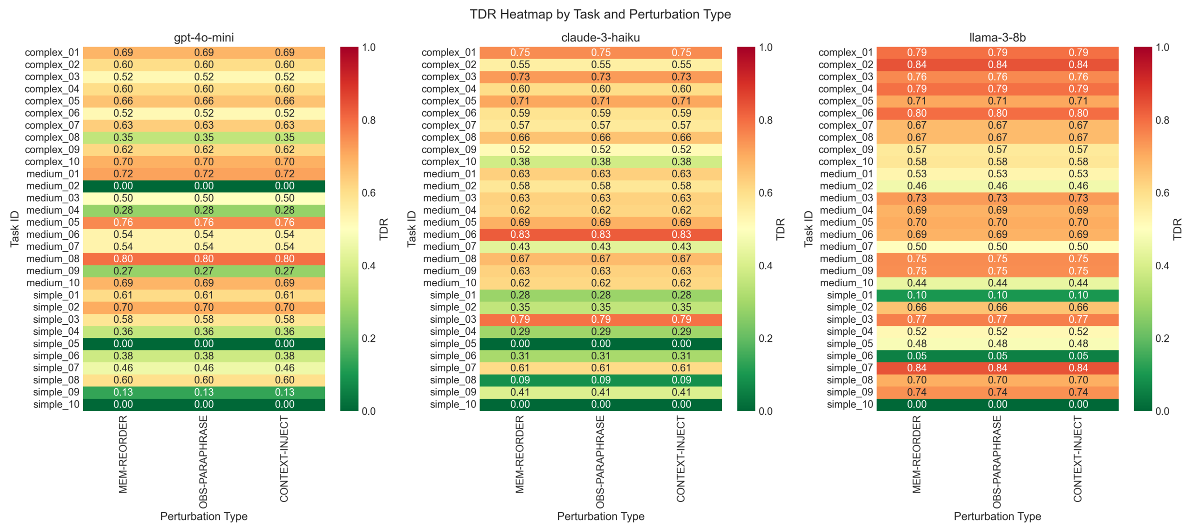

Figure 6: Mean TDR by task and model. Tasks vary dramatically in their vulnerability to perturbations, ranging from near-perfect stability to near-complete divergence. Some tasks show consistent instability across all models, while others exhibit model-specific patterns.

5.8 Robustness Checks

To ensure our findings are not artifacts of task failures, we performed robustness analyses:

- **Excluding consistently failing tasks**: TDR decreases by only 7.1% (from 0.54 to 0.46), confirming instability is not driven by failure-related noise.

- **DOS > 1 for all trajectories**: No trajectory diverged at step 1, ruling out immediate parsing failures.
- **6 tasks show perfect stability** (TDR $= 0.00$ across all models), demonstrating that stability is achievable for some task structures.

6 Discussion

6.1 Success $\neq$ Reliability $\neq$ Auditability

Our central finding, that agents can succeed reliably while behaving unreliably, has important implications. Current evaluation practices that report only success rate may mask significant behavioral inconsistency. Two agents with identical success rates may differ dramatically in trajectory stability, with consequences for:

- **Debugging**: Inconsistent behavior makes failure analysis difficult.
- **Monitoring**: Operators cannot establish behavioral baselines.
- **Safety**: Unpredictable paths may encounter unanticipated hazards.

6.2 Cross-Model Patterns

The consistency of instability across three different model families (OpenAI, Anthropic, Meta) suggests this is a fundamental property of current LLM architectures, not specific to any vendor or training approach. The ranking (GPT-4o-mini most stable, Llama-3-8B least stable) may reflect differences in model scale, training data, or RLHF procedures, but all models exhibit substantial instability.

6.3 Implications for Agent Deployment

The finding that perturbations can *improve* outcomes is particularly concerning for deployment. It suggests that agent behavior exists on a knife's edge, where minor input variations can tip outcomes either way. This argues for:

- **Ensemble methods**: Running multiple perturbed versions and aggregating.
- **Stability regularization**: Training or prompting for consistent behavior.
- **Conservative deployment**: Avoiding high-stakes applications until stability improves.

6.4 Why Does This Happen?

We hypothesize several contributing factors:

Soft attention over context LLMs process all input tokens simultaneously via attention. Reordering memory or injecting context changes attention patterns, potentially surfacing different information for decision-making.

No explicit planning Unlike classical planners, LLM agents do not maintain explicit goal representations or search trees. Each action is generated independently conditioned on context, making the trajectory sensitive to context variations.

Underspecified training objective LLMs are trained to predict text, not to exhibit stable policies. There is no training signal for behavioral consistency.

We interpret the observation that perturbations can improve outcomes through the lens of **stochastic policy basins**. LLM agents do not execute explicit plans but sample actions from high-dimensional conditional distributions over tokens. A given prompt may place the model in a low-probability basin corresponding to a suboptimal action sequence. Semantics-preserving perturbations can shift the model into an alternative basin with higher expected return, analogous to noise-assisted exploration in non-convex optimization. From this perspective, beneficial perturbations are not evidence of flawed prompting but a consequence of inherently unstable policy landscapes.

7 Limitations

Model Selection While we evaluated three models from different families, other models (GPT-4, Claude-3-opus, Llama-3-70B) may exhibit different stability profiles. Larger models might show improved stability.

Synthetic Environment FileWorld is intentionally simple. Real-world environments with richer observations and larger action spaces may show different patterns.

API Fingerprint Drift Despite deterministic settings, cloud-hosted models showed fingerprint variation, indicating some non-determinism from backend changes. We mitigate this by logging all fingerprints.

Task Coverage Thirty tasks provide substantial evidence but cannot fully characterize stability across the space of possible tasks.

No Human Comparison We do not compare to human trajectory stability, which would provide a natural baseline for "acceptable" variation.

8 Conclusion

We have presented a systematic study of trajectory stability in LLM-based agents across three model families, introducing metrics (TDR, DOS) and a controlled perturbation methodology for measuring behavioral consistency. Our experiments with GPT-4o-mini, Claude-3-haiku, and Llama-3-8B reveal substantial trajectory instability across all models—approximately half of actions differ under semantics-preserving perturbations—despite maintained task success. Most strikingly, perturbations can improve outcomes equally as often as they harm them, suggesting agents occupy unstable policy basins.

These findings argue for trajectory stability as a first-class evaluation criterion alongside task success. We release our experimental infrastructure to enable reproducible stability measurement for future agent systems.

Broader Impact Understanding agent stability is essential for safe deployment. Our work highlights risks but does not directly enable harm. We hope these findings encourage the development of more predictable and trustworthy LLM agents.

References

Liu, X., Yu, H., Zhang, H., Xu, Y., Lei, X., Lai, H., ... & Dong, Y. (2023). AgentBench: Evaluating LLMs as Agents. *arXiv preprint arXiv:2308.03688.*

Lu, Y., Bartolo, M., Moore, A., Riedel, S., & Stenetorp, P. (2022). Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *ACL 2022.*

Perez, F., & Ribeiro, I. (2022). Ignore this title and HackAPrompt: Exposing systemic vulnerabilities of LLMs through a global scale prompt hacking competition. *arXiv preprint arXiv:2311.16119.*

Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Zettlemoyer, L., ... & Scialom, T. (2023). Toolformer: Language models can teach themselves to use tools. *NeurIPS 2023.*

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347.*

Yang, J., Jimenez, C. E., Wettig, A., Liber, K., Narasimhan, K., & Press, O. (2024). SWE-agent: Agent-computer interfaces enable automated software engineering. *arXiv preprint arXiv:2405.15793.*

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2022). ReAct: Synergizing reasoning and acting in language models. *ICLR 2023.*

Zhao, Z., Wallace, E., Feng, S., Klein, D., & Singh, S. (2021). Calibrate before use: Improving few-shot performance of language models. *ICML 2021.*

Zhou, S., Xu, F. F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., ... & Neubig, G. (2023). WebArena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854.*

A Generation Configuration

To maximize reproducibility of the behavioral patterns observed, all experiments were conducted using the following generation configurations:

- **GPT-4o-mini**: `temperature=0`, `top_p=1`, `seed=42`
- **Claude-3-haiku**: `temperature=0`, `top_p=1`
- **Llama-3-8B**: Greedy decoding (`do_sample=False`)
- Max tokens: 512
- Max steps: 20

B FileWorld Environment Details

The environment consists of 10 nodes connected by directed edges. Some edges require items (e.g., key_card for archive access). The observation format is:

```
=== CURRENT STATE ===
Location: {node_name}
Description: {description}

Visible Items: {sorted_items}
Inventory: {sorted_inventory}

=== AVAILABLE ACTIONS ===
- go {location}: Move to {location}
- pickup {item}: Take {item}
...
```

C Complete Model Statistics

Table 4: Complete statistics for all 1,440 trajectories.

Metric	**GPT-4o-mini**	**Claude-3-haiku**	**Llama-3-8B**
Total Trajectories	480	480	480
Tasks	30	30	30
Baseline Success Rate	46.7%	63.3%	53.3%
Perturbed Success Rate	47.3%	63.6%	53.6%
Mean TDR	0.493	0.517	0.610
Std TDR	0.228	0.220	0.221
Mean DOS	5.66	4.96	3.94
Std DOS	1.99	1.59	1.26
Cohen's d	2.17	2.35	2.76
p-value	1.2×10^{-12}	1.7×10^{-13}	2.7×10^{-15}
FS Flips	29	43	40
SF Flips	26	42	39
SS (Stable Success)	184	243	201
FF (Stable Failure)	211	122	170

D Step Count Analysis

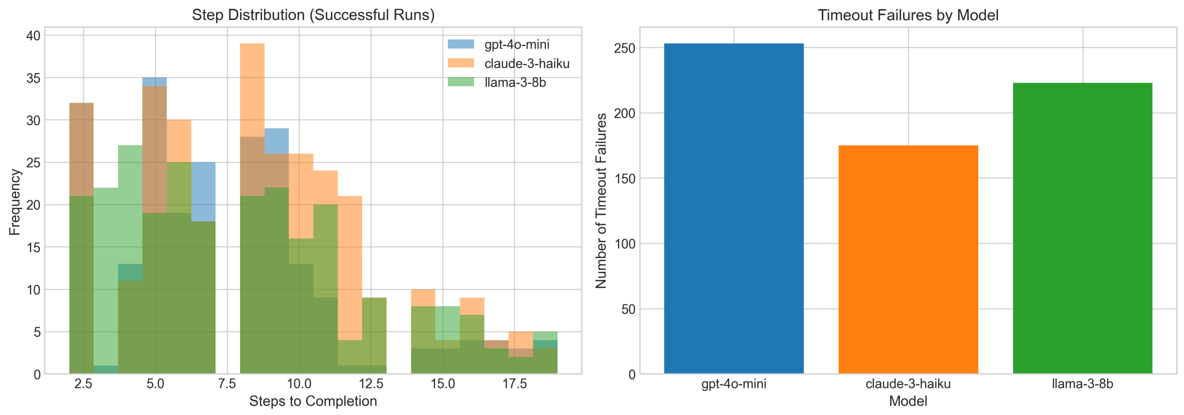

Figure 7: Step count distributions across all models. Left: Distribution of steps to completion for successful runs. Right: Number of failed runs (hitting 20-step timeout) by condition.