Agent Planning Benchmark: A Diagnostic Framework for Planning Capabilities in LLM Agents

Haoyu Sun[1,2,*] **Wenxuan Wang**[3,2,*] **Mingyang Song**[4] **Jujie He**[5]
Weinan Zhang[3] **Yang Liu**[6] **Yang Yang**[7,†] **Yu Cheng**[8,2,†]

[1]Tongji University [2]Shanghai AI Laboratory [3]Harbin Institute of Technology
[4]Fudan University [5]Skywork AI [6]University of California, Santa Cruz
[7]Shanghai Jiao Tong University [8]Nanyang Technological University

[*]Equal contribution. [†]Corresponding authors.

Abstract

Planning is central to LLM agents: before acting, an agent must decompose goals, select tools, reason over constraints, and decide when a task is infeasible. Yet existing agent evaluations often report only end-to-end success, making it difficult to determine whether failures stem from planning or execution. We introduce **Agent Planning Benchmark (APB)**, a planning-specific diagnostic benchmark with 4,209 multimodal cases across 22 domains and five settings, covering holistic planning, feedback-conditioned step-wise planning, and robustness under extraneous tools, broken tools, and unsolvable tasks. Across 12 MLLMs, APB reveals systematic weaknesses in long-horizon planning, tool-noise robustness, calibrated refusal, and inference-time refinement. We further validate APB on 200 ToolSandbox tasks and 200 τ^2-bench tasks, where APB-guided refinement consistently improves plan correctness, plan grade, and downstream execution metrics across three representative models. APB thus serves as an upstream diagnostic complement to execution benchmarks. The APB benchmark and code are available in this URL.

1 Introduction

Large Language Models (LLMs) have rapidly evolved from passive text generators into agents that plan, call tools, and interact with digital environments. In successful agent systems, planning is not an auxiliary behavior but the organizing layer that determines how goals are decomposed, which tools are selected, how intermediate evidence is used, and when the agent should stop or refuse. This role is reflected in ReAct and Reflexion, where reasoning traces and feedback-driven correction guide action (Yao et al., 2022; Shinn et al., 2023); in open-ended agents such as Voyager (Wang et al., 2023a); and in workflow-level

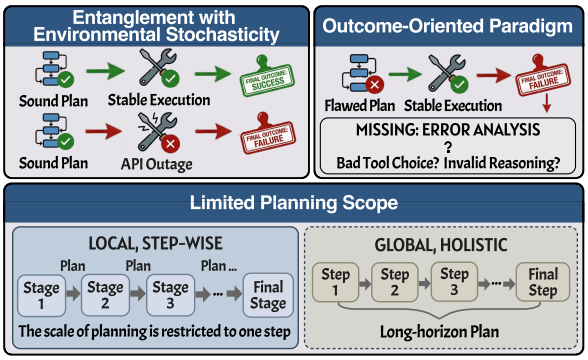

Figure 1: Systematic limitations in existing agent planning benchmarks.

systems such as MetaGPT, LLMCompiler, and SWE-agent (Hong et al., 2024; Kim et al., 2024; Yang et al., 2024). These systems demonstrate that planning is a foundational capability for reliable agents, shaping both high-level task decomposition and downstream tool-use behavior.

Despite this importance, current evaluations do not yet provide a sufficiently diagnostic view of planning. End-to-end benchmarks are indispensable for deployed behavior, but final success entangles plan quality, tool invocation, environmental instability, and recovery (Liu et al., 2024; Deng et al., 2023). As Figure 1 illustrates, such outcome-oriented evaluation often leaves failure causes ambiguous. Meanwhile, planning-oriented benchmarks frequently focus on narrow or static formulations, leaving limited coverage of global long-horizon planning, feedback-conditioned local planning, noisy tool spaces, and feasibility assessment under missing information.

To address this gap, we introduce the **Agent Planning Benchmark (APB)**, a planning-focused diagnostic benchmark comprising 4,209 multimodal test cases across 22 categories. APB evaluates planning at multiple granularities: **Holistic Planning** asks models to produce complete plans

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 22008–22070
October 24-29, 2026 ©2026 Association for Computational Linguistics

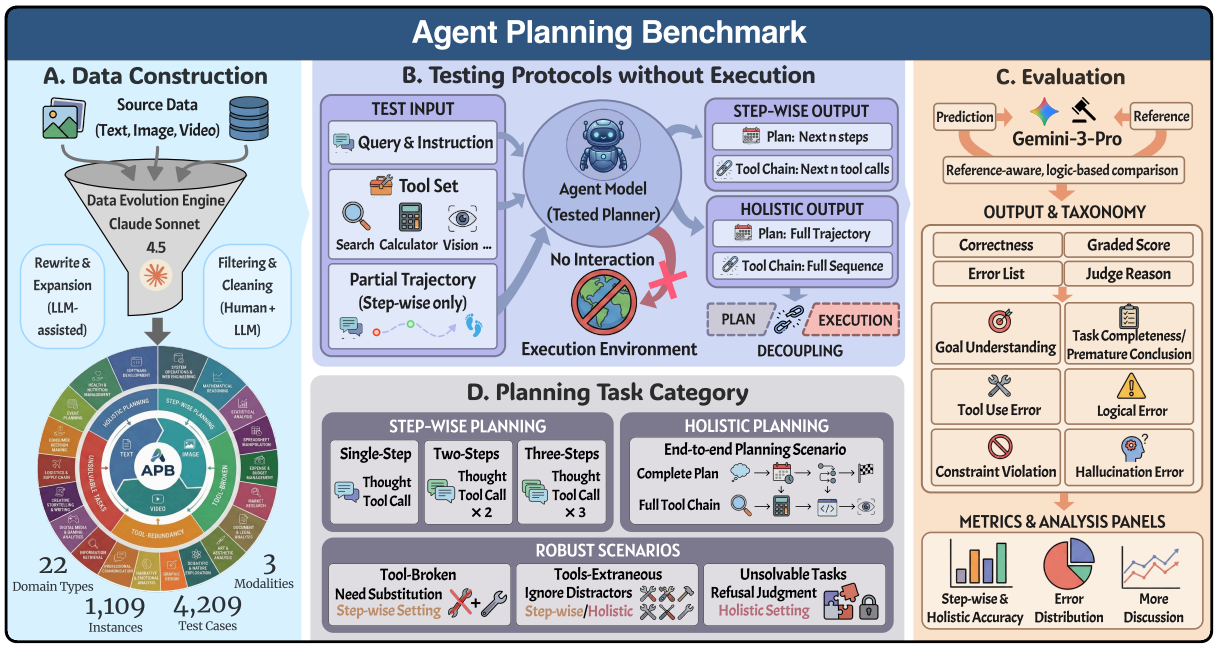

Figure 2: **Overview of APB.** The framework comprises: **(A) Data Construction:** A pipeline for synthesizing complex planning instances via evolution and filtering. **(B) Planning Protocols:** Holistic and feedback-conditioned step-wise tasks for evaluating planning logic. **(C) LLM-as-Judge:** Automated logic-based assessment providing a comprehensive error taxonomy. **(D) Task Categories:** Five complementary tasks for comprehensive assessment.

and tool chains for long-horizon tasks, while **Step-wise Planning** conditions models on partial execution trajectories and tool-return feedback. APB further stresses robustness through extraneous tools, broken tools, and unsolvable tasks. Rather than only asking whether an agent succeeds, APB diagnoses why a plan succeeds or fails through Plan Correctness, Plan Grade, and an E1–E6 error taxonomy. Our executable validation further examines APB's practical signal in controlled environments. On 200 ToolSandbox (Lu et al., 2025) and 200 τ^2-bench (Barres et al., 2025) tasks, we compare direct execution, plan-first execution, and APB-guided refinement across GPT-4o, Qwen3-VL-235B-A22B, and Gemini 2.5 Flash. Refined plans consistently improve downstream execution metrics, supporting APB as an upstream diagnostic signal connected to executable agent behavior.

Our evaluation across 12 Multimodal Large Language Models (MLLMs) reveals substantial variation in planning capability. Newer proprietary models dominate long-horizon holistic planning, while open-source systems remain fragile under tool noise and feasibility constraints. We further find that inference-time refinement is highly effective for holistic planning, whereas short-horizon step-wise decisions benefit less from extended reflection and may suffer from over-correction. These findings show that planning quality is not a monolithic capability: it differs across horizons, feedback conditions, and robustness settings. To summarize, our contributions are as follows:

- We propose APB, a planning-specific diagnostic benchmark with 4,209 multimodal cases across holistic, feedback-conditioned step-wise, and robustness-oriented planning tasks.
- We design a hierarchical evaluation framework with Plan Correctness, Plan Grade, and a human-informed E1–E6 error taxonomy, enabling fine-grained root-cause analysis.
- We evaluate 12 MLLMs and identify systematic weaknesses across horizons, feedback conditions, and robustness settings.
- We validate APB-guided refinement on ToolSandbox and τ^2-bench, showing gains in both planning and execution metrics.

2 Related Work

Planning in LLM-based Agents The paradigm of autonomous agents has evolved from simple reasoning loops to systems capable of multi-step planning and environmental interaction (Xi et al., 2025; Luo et al., 2025; Durante et al., 2024; Li et al., 2024). Foundational frameworks such as ReAct (Yao et al., 2022) and Reflexion (Shinn et al., 2023) bind reasoning, action selection, and feedback-based revision, showing that plans and

intermediate reasoning traces are central to agent behavior. Planning-centric systems further rely on task decomposition, memory, scheduling, and agent-computer interfaces to improve downstream execution, as in Voyager, MetaGPT, LLMCompiler, and SWE-agent (Wang et al., 2023a; Hong et al., 2024; Kim et al., 2024; Yang et al., 2024). These lines of work motivate direct evaluation of planning: without coherent goals, constraints, and tool-use strategies, downstream execution has little chance of succeeding reliably.

Agent Benchmarks Evaluating agentic capabilities has progressed from general-purpose assessments (Liu et al., 2024; Chang et al., 2024) to rigorous environment-specific testbeds. In digital domains, benchmarks span web browsing (Zhou et al., 2024), operating systems (Xie et al., 2024b; Bonatti et al., 2025), and mobile interfaces (Deng et al., 2024). Concurrent efforts target specific skills, including tool utilization (Wang et al., 2024; Huang et al., 2024; Shen et al., 2024; Ye et al., 2025), code generation (Yang et al., 2025a; Si et al., 2025), and scientific reasoning (Wang et al., 2023b), alongside embodied interaction benchmarks (Shridhar et al., 2020; Liu et al., 2025). PlanCraft evaluates long-horizon, multimodal, and deliberately unsolvable planning problems in the Minecraft crafting domain (Dagan et al., 2024). Recently, attention has narrowed toward the cognitive core of agents, especially planning and reasoning capabilities (Li et al., 2025; Wei et al., 2025). APB complements these efforts through its combined coverage of 22 application domains, text/image/video inputs, holistic and feedback-conditioned step-wise planning, robustness to extraneous and broken tools in addition to unsolvable tasks, and E1–E6 diagnostic evaluation. This design provides a diagnostic view that end-to-end benchmarks cannot offer, while our executable validation examines whether APB-guided planning improvements translate to controlled execution settings.

3 Agent Planning Benchmark

3.1 Task Definition

APB defines two core tasks: **Holistic** and **Step-wise Planning**. To systematically assess planning robustness under realistic tool-use conditions, we further introduce three adversarial variants that test tool selection under noisy environments, recovery from broken tools, and principled refusal.

Holistic Planning (1,109 instances) Holistic planning evaluates whether a model can construct a complete solution plan before acting. Given a query Q, a system prompt S, and a tool set $\mathcal{T}$, the tested model is required to produce a complete solution path in a single pass: $f_{\text{holistic}} : (Q, S, \mathcal{T}) \rightarrow (P, \mathcal{C})$. Specifically, the model jointly generates:

- **Plan** P: A high-level decomposition translating abstract goals into logically coherent steps;
- **Tool Chain** $\mathcal{C} = [e_1, e_2, \ldots, e_m]$: a sequence of m tool invocation nodes, with each node $e_i = (t_i, \theta_i, r_i)$ comprising tool identifier t_i, invocation arguments θ_i, and explicit rationale r_i.

Step-wise Planning (900 instances) In contrast, effective agent behavior also requires localized reasoning conditioned on evolving execution contexts and prior tool feedback.

Given a system prompt S, a tool set $\mathcal{T}$, and a partially executed trajectory $\tau_{1:j}$ that includes the initial query, previous actions, and observed tool returns, the model is tasked with predicting the next k actions: $f_{\text{step-wise}} : (S, \mathcal{T}, \tau_{1:j}) \rightarrow (a_{j+1}, a_{j+2}, \ldots, a_{j+k})$. Each action $a_i = (c_i, t_i, \theta_i)$ comprises dialogue content c_i and a tool invocation (t_i, θ_i). According to the prediction horizon k, we further distinguish:

- **Single-step Planning** ($k = 1$): predicting only the immediate next action;
- **Multi-step Planning** ($k \in \{2, 3\}$): predicting the subsequent two or three actions.

This design targets feedback-conditioned planning: the model uses execution history to choose its next action while avoiding premature conclusions.

Tool-Extraneous Planning (1,500 instances) To simulate open-domain noise, this task injects n_e semantically similar but irrelevant tools $\mathcal{T}_r$ into $\mathcal{T}$, forming $\mathcal{T}' = \mathcal{T} \cup \mathcal{T}_r$, where $n_e \in \{2, 4, 6, 8, 10\}$: $f_{\text{extraneous}} : (Q, S, \mathcal{T}') \rightarrow (P, \mathcal{C})$. The task contains 1,500 cases expanded from 150 holistic and 150 step-wise instances.

Tool-Broken Planning (300 instances) This task evaluates recovery from tool failures. Based on step-wise planning, we replace the return value of a critical tool t_j with an erroneous output o_j^{error}, while introducing an alternative tool t'_j: $f_{\text{broken}} : (S, \mathcal{T} \cup \{t'_j\}, \tau_{1:j-1}, o_j^{\text{error}}) \rightarrow (a_{j+1})$.

Unsolvable Planning (400 instances) To evaluate principled refusal, we construct logically unsolvable instances that require $f_{\text{unsolvable}}$:

$(Q, S, \mathcal{T}) \rightarrow$ `REJECT`. We derive 400 cases from 100 holistic instances across contradictory constraints, missing information, inaccessible visual evidence, and tool removal (Appendix B.4.3).

3.2 Data Construction

We aggregate data from OpenCUA, GTA, GAIA, ToolBench, FrameThinker, and real traffic from a public online AI agent platform (Wang et al., 2025b, 2024; Mialon et al., 2024; Qin et al., 2024; He et al., 2025). Each instance contains a system prompt, user query, tool set, and execution trajectory. We expand limited seed tasks by enriching queries, expanding tools, and simulating longer trajectories with Claude Sonnet 4.5 (Anthropic, 2025). Rule-based checks, two-stage LLM validation, and human verification ensure quality; holistic ground truth is then synthesized from validated trajectories. Details are in Appendix B.

3.3 Metrics

We evaluate performance across three dimensions: correctness, severity, and error typology.

(1) Plan Correctness This binary metric serves as a strict indicator of planning success, determining whether the generated plan is logically sound, executable, and fully satisfies the task objectives.

(2) Plan Grade Plan Grade measures the severity of planning deficiencies against the task requirements rather than distance from a unique or optimal reference path. By evaluating goal attainment, completeness, constraint satisfaction, and logical consistency, this metric differentiates between near-misses and fundamental failures, utilizing a discrete scale of $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$.

(3) Error Taxonomy

To systematically diagnose failure mechanisms in agent planning, we define a unified taxonomy with **six** semantically independent, annotatable error categories, where a single sample may exhibit multiple errors simultaneously. The taxonomy was developed through iterative human inspection of planning failures, assisted by LLM-based clustering and summarization, and then finalized through additional rounds of human checks to ensure that each category is both interpretable and consistently annotatable. Notably, E2 instantiates two protocol-specific subtypes that capture a key behavioral distinction between step-wise and holistic planning.

- **E1: Goal Understanding Error** Misinterpreting the fundamental user intent or task objective.
- **E2: Premature Conclusion / Task Incompleteness** Critical failures in plan termination. The former denotes stopping before the task is fully solved (step-wise), while the latter indicates a failure to cover all required sub-tasks in the generated trajectory (holistic).
- **E3: Constraint Violation** Violating explicit system prompt instructions or task constraints.
- **E4: Logic Error** Invalid causal dependencies, illogical ordering, missing prerequisites, or circular reasoning. Deviation from the reference path alone is not an E4 error when the alternative remains feasible and satisfies the task constraints (Appendix D.2).
- **E5: Tool Use Error** Misunderstanding intended tool functionality or operational semantics.
- **E6: Hallucination** Generating unsupported fabricated facts or hallucinated intermediate results.

This hierarchical framework enables both quantitative benchmarking and root-cause diagnosis: Correctness signals planning success, Grade quantifies severity, and Taxonomy identifies failure mechanisms (see Appendix C.2). In practice, low APB scores indicate intrinsic planning deficits such as missing constraints or invalid tool choices, while high APB scores paired with low execution success can help localize bottlenecks to execution-side engineering. We operationalize the metrics via an LLM-as-Judge suite to validate logical coherence against reference trajectories (see Appendix D).

4 Experiments

4.1 Evaluation Settings

We evaluate 12 SOTA MLLMs on APB, including six open-source models (the Qwen3VL series: Qwen3VL-30B-A3B-Instruct, Qwen3VL-32B-Instruct, and Qwen3VL-235B-A22B-Instruct (Qwen Team, 2025); and the InternVL3.5 series: InternVL3.5-30B-A3B, InternVL3.5-38B, and InternVL3.5-241B-A28B (Wang et al., 2025a)) and six proprietary models (Claude Sonnet 4.5 (Anthropic, 2025), GPT-5 (OpenAI, 2025), GPT-4o (OpenAI, 2024), Gemini 3 Pro (DeepMind, 2025), and Gemini 2.5 series (Pro/Flash) (Team, 2025)).

4.2 Main Results

4.2.1 Overall Performance

Table 1 shows clear divergence across planning modes. All models drop from feedback-conditioned step-wise planning to holistic planning,

Model	Holistic Planning								Tool-Extraneous								
	CR	Grade	E1	E2	E3	E4	E5	E6	CR	Grade	E1	E2-H	E2-S	E3	E4	E5	E6
GPT-5	**74.5**	**0.91**	0.3	2.9	16.4	6.3	2.3	3.8	67.0	0.81	4.7	0.9	9.0	8.0	8.6	15.8	1.9
Gemini 3 Pro	71.3	0.89	1.1	6.5	16.7	6.9	4.7	2.7	**76.4**	0.87	3.2	1.1	7.4	8.0	7.6	8.8	1.5
Claude Sonnet 4.5	64.1	0.86	1.0	4.5	24.4	11.4	3.2	5.3	76.4	**0.88**	2.4	1.9	4.8	10.7	7.8	6.7	2.7
Gemini 2.5 Pro	55.0	0.81	1.1	9.3	26.4	15.2	5.2	10.7	66.8	0.81	3.7	3.1	11.5	13.7	10.9	10.0	4.0
Gemini 2.5 Flash	36.5	0.69	1.5	31.9	28.0	17.5	6.3	10.2	59.8	0.79	3.5	**4.4**	28.4	14.5	13.1	7.3	4.4
GPT-4o	19.5	0.62	1.7	33.5	43.7	36.4	15.2	9.3	45.5	0.70	4.9	3.3	34.2	20.3	24.9	18.4	4.6
InternVL3.5-241B-A28B	18.1	0.62	2.5	25.9	46.6	42.0	15.2	12.1	48.7	0.73	3.5	1.7	26.2	19.7	27.3	15.5	6.2
InternVL3.5-30B-A3B	8.4	0.49	**7.8**	**47.9**	49.7	**45.1**	**25.6**	15.1	33.4	0.60	**7.8**	3.2	**49.3**	23.5	**37.1**	**24.5**	8.9
InternVL3.5-38B	16.2	0.61	2.3	25.5	46.4	43.2	21.1	17.0	45.3	0.71	4.0	3.1	24.7	19.5	29.1	19.5	7.8
Qwen3VL-235B-A22B-In.	21.8	0.64	3.0	17.8	44.6	32.7	8.8	25.5	52.6	0.74	3.8	2.8	15.6	19.6	21.6	12.5	11.5
Qwen3VL-30B-A3B-In.	10.7	0.55	3.1	26.3	**50.0**	38.1	17.6	**32.8**	40.2	0.67	4.7	**4.4**	29.4	**23.7**	26.1	18.9	**15.9**
Qwen3VL-32B-In.	22.5	0.64	1.5	17.0	48.2	28.5	13.5	24.6	51.9	0.73	3.6	2.1	12.5	17.5	17.6	9.2	9.2

Model	Step-wise Planning								Tool-Broken					Unsolvable			
	CR	Grade	E1	E2	E3	E4	E5	E6	Rp	Al	Rt	Rf	Ot	Conf	Info	Tool	Vis
GPT-5	77.4	0.85	3.9	0.8	5.9	16.3	8.2	**0.0**	76.3	17.3	5.7	0.0	0.7	**80**	40	**75**	**89**
Gemini 3 Pro	**80.1**	**0.87**	4.2	0.9	4.3	12.7	7.8	0.3	**78.0**	18.3	0.7	0.0	3.0	65	43	65	16
Claude Sonnet 4.5	77.3	0.86	3.0	1.6	3.9	16.2	7.0	1.7	76.7	14.7	4.3	0.3	4.0	72	19	16	53
Gemini 2.5 Pro	69.2	0.79	4.4	2.0	9.3	18.0	10.6	**2.6**	74.7	12.3	10.0	1.0	2.0	77	**65**	**75**	11
Gemini 2.5 Flash	66.6	0.78	5.6	4.4	9.0	21.4	12.7	2.2	44.3	28.3	**24.3**	**2.0**	1.0	74	56	72	36
GPT-4o	57.6	0.73	5.1	2.7	7.4	31.3	16.7	2.1	42.3	**46.3**	6.3	0.3	4.7	78	39	30	64
InternVL3.5-241B-A28B	59.2	0.74	6.6	2.0	7.0	30.3	16.1	1.4	74.7	19.0	4.0	0.0	2.3	75	39	38	38
InternVL3.5-30B-A3B	38.8	0.58	**9.0**	2.0	**20.3**	**44.3**	**27.0**	1.3	37.7	38.3	15.7	0.0	**8.3**	69	35	18	50
InternVL3.5-38B	49.7	0.68	7.1	2.1	7.4	38.9	21.6	1.1	57.0	29.0	9.0	0.7	4.3	60	6	10	20
Qwen3VL-235B-A22B-In.	60.2	0.74	6.4	1.4	7.8	30.1	13.3	2.4	77.7	16.7	4.3	0.0	1.3	72	11	7	64
Qwen3VL-30B-A3B-In.	49.6	0.68	8.0	3.2	9.9	38.4	19.1	1.3	55.7	33.3	7.0	0.0	4.0	61	10	2	44
Qwen3VL-32B-In.	59.9	0.75	5.6	1.2	8.4	29.0	15.1	0.8	68.7	22.0	7.3	0.0	2.0	76	31	29	84

Table 1: **Overall APB results.** CR denotes Correctness Rate; Grade is average Plan Grade. E1–E6 report error incidence rates. Tool-Broken columns show behavior distributions, and Unsolvable columns show refusal rates across four infeasibility types.

Model	Holistic				Step-wise			
	TC	CR_B	CR_E	Δ	TC	CR_B	CR_E	Δ
GPT-5	**115**	77.3	55.4	**-22.0**	23	85.3	78.7	**-6.7**
Gemini 3 Pro	46	77.2	69.9	**-7.3**	12	80.0	82.9	+2.9
Claude Sonnet 4.5	**50**	79.3	67.0	**-12.3**	6	86.7	85.7	-0.9
Gemini 2.5 Pro	**60**	61.3	54.7	**-6.6**	7	80.0	78.9	-1.1
Gemini 2.5 Flash	27	53.3	40.2	**-13.1**	13	78.0	79.3	+1.3
GPT-4o	**77**	26.7	21.4	**-5.3**	15	73.3	69.6	-3.7
InternVL3.5-241B	**80**	22.3	23.4	+1.1	15	77.3	74.0	-3.3
InternVL3.5-30B	**78**	12.0	8.9	-3.1	6	60.0	57.9	-2.1
InternVL3.5-38B	**98**	24.0	25.2	+1.2	9	67.3	65.3	-2.0
Qwen3VL-235B	**73**	30.7	29.7	-1.0	5	74.0	75.5	+1.5
Qwen3VL-30B	**97**	16.0	12.7	-3.3	13	64.0	67.7	+3.7
Qwen3VL-32B	**108**	32.7	31.2	-1.5	21	72.0	72.5	+0.5

Table 2: **Impact of extraneous tools.** TC is extraneous-tool usage frequency; CR_B/CR_E are correctness rates in base and extraneous settings.

with open-source models particularly vulnerable to long-horizon global reasoning. This gap indicates that complete tool-use strategies require more than locally reasonable next-step decisions. Notably, the dense Qwen3-VL-32B-Instruct matches or outperforms much larger MoE models in both holistic and step-wise tasks. Figure 3 further shows that Logic Error (E4) is a shared bottleneck, while holistic planning also suffers from incompleteness, constraint violations, and hallucination. Step-wise planning instead concentrates failures around local tool use and logic, reflecting its narrower decision scope and access to prior feedback.

This contrast is central to APB's design. Holistic planning exposes whether a model can maintain global consistency across a complete plan, while step-wise planning asks whether it can use the current trajectory state to choose the next immediate action. The aggregate model rankings are strongly correlated, yet a matched evaluation on the same 300 underlying samples reveals a systematic within-model difficulty gap between the two settings; full paired results and significance tests are reported in Appendix G.1.

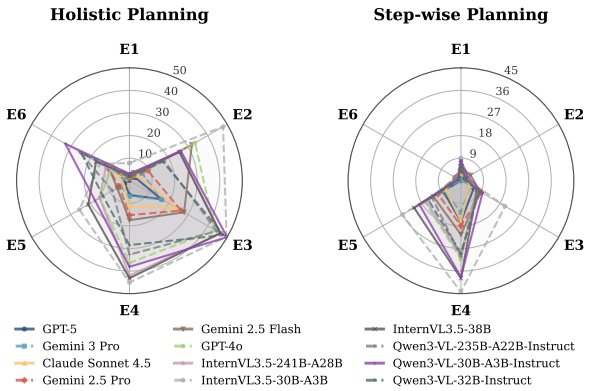

Figure 3: Comparative error distributions across planning modes.

4.2.2 Analysis of Extraneous Tools

Introducing extraneous tools significantly degrades holistic planning performance, whereas step-wise planning remains resilient. As detailed in Table 2, extraneous tools cause broad performance degrada-

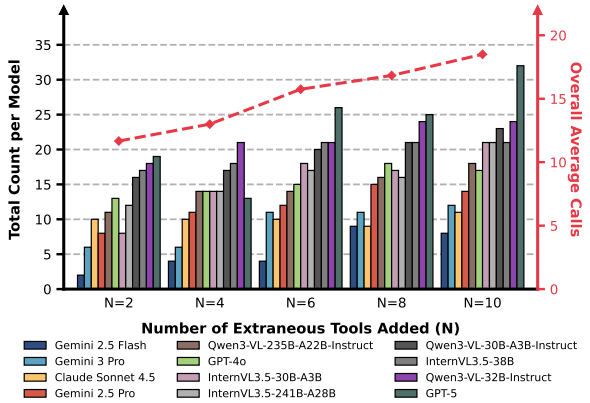

Figure 4: Trends in extraneous tool usage. The red dashed line indicates the overall average count.

tion: top-tier models such as GPT-5 and Claude Sonnet 4.5 exhibit substantial holistic accuracy declines of 22.0% and 12.3%, in contrast to the much smaller 0.9%–6.7% drops in step-wise mode. This vulnerability is further corroborated by Figure 4, which illustrates a positive correlation between extraneous-tool availability and erroneous utilization within holistic planning trajectories.

This disparity highlights distinct paradigmatic vulnerabilities: holistic planning requires one-shot trajectory generation, where expanding the tool library amplifies the cumulative probability of selection errors, leading to cascading downstream plan failures. Conversely, step-wise planning relies on isolated, locally optimal decisions, rendering it comparatively more resilient to noise interference.

4.2.3 Adaptability to Broken Tool Setting

In the Tool-Broken task, we classify model behaviors into five distinct categories: **Replace** (switching to the designated substitute tool), **Alter** (using a non-substitute tool for a different action), **Retry** (reattempting the same tool), **Refuse** (declining to act), and **Other**. Among these, **Replace** is regarded as the optimal behavior, indicating successful recovery by switching tools to continue the task.

Our analysis reveals different behavioral patterns. Gemini 3 Pro and Qwen3VL-235B-A22B-Instruct demonstrate superior adaptability with high Replace rates (>77%), underscoring their proficiency in identifying valid substitutes. Conversely, GPT-4o and Gemini 2.5 Flash record lower Replace proportions (<45%), diverging significantly into Alter or Retry actions, respectively. This indicates that robust planners favor adaptive replacement, whereas other models struggle with uncertainty, resorting to divergent or repetitive behaviors.

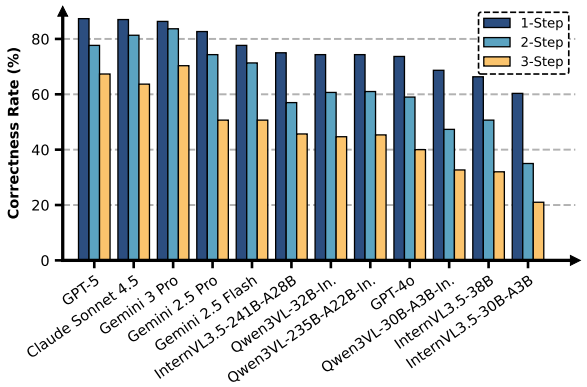

Figure 5: Step-wise Planning performance across varying prediction horizons.

4.2.4 Awareness of Unsolvable Tasks

As shown in Table 1, GPT-5 demonstrates superior robustness, yet reveals a general gap where models handle explicit Constraint Conflict far better than Information Missing. This disparity suggests that current models can readily detect explicit logical contradictions stated in system prompts or instructions, but struggle with more implicit and non-explicitly stated missing information, often defaulting to hallucination rather than appropriate refusal. Moreover, open-source models notably underperform on Tool Removal, exposing deficiencies in feasibility assessment where they attempt tasks blindly despite insufficient toolsets. Even proprietary models remain vulnerable: Claude Sonnet 4.5 performs poorly on Tool Removal, while the Gemini series is notably weaker on Visual Missing scenarios. These failures highlight a persistent tendency to rely on implicit assumptions and over-completion when critical evidence is absent, rather than exercising calibrated refusal.

5 Discussion

5.1 Executable Validation of APB-Guided Planning

To test whether APB's diagnostic signals translate into executable agent behavior, we validate on Tool-Sandbox and τ^2-bench. For each environment, we evaluate 200 tasks under three controlled execution modes. **Direct** executes the task without planning. **Plan-first** generates an initial holistic plan p_0 and then executes it. **Refine** starts from the same p_0, applies one round of APB-guided critique, and executes the revised plan p_1. Gemini 3 Pro produces only Plan Grade, E1–E6 indicators, and a diagnostic rationale under an anti-leak instruction; the tested model then revises its own plan without ac-

cess to hidden supervision. Plan-first and Refine otherwise use identical executors, sampling configurations, and environments, so their comparison isolates the revision effect. ToolSandbox reports average trajectory similarity (higher is better), while τ^2-bench reports average task reward (higher is better). For APB metrics, "$a \rightarrow b$" denotes the score before and after APB-guided refinement. Appendix C.10 specifies the environment-specific supervision and complete single-round protocol.

Table 3 shows consistent gains from APB-guided refinement. Relative to Plan-first execution of p_0, executing p_1 improves ToolSandbox similarity by 3.4–5.5 points and τ^2-bench reward by 0.03–0.07 across the three models. Since Plan-first and Refine use the same initial draft plan and differ only in whether APB-guided critique is applied, these results demonstrate that APB provides actionable diagnostic signals for identifying and correcting planning defects that affect executable behavior.

5.2 Dynamics of Step-wise Planning: Horizon and Temporal Profiles

> ***Takeaway 1.*** Extended planning horizons accumulate errors; distinct models exhibit diverging performance peaks across temporal stages.

To investigate the impact of planning horizons and historical context, we analyze performance disparities under varying prediction scopes and temporal truncation points. As illustrated in Figure 5, extending the prediction scope from 1-Step to 3-Step causes a monotonic decline in correctness across all models, highlighting the challenge of predicting multiple future actions even when prior tool feedback is available. Further analysis of interaction trajectories in Figure 6 reveals divergent patterns: models such as Gemini 2.5 series and most open-source variants peak early and degrade as context increases, suggesting limitations in leveraging extended feedback histories. Conversely, models like GPT-5 reach peak performance at intermediate stages, demonstrating the ability to use accumulated context for decision optimization. These findings show why APB includes feedback-conditioned step-wise tasks in addition to holistic planning.

This analysis distinguishes two feedback-related skills. The first is reactive adaptation: using the latest observation to choose the immediate next action. The second is look-ahead consistency: projecting several future actions while preserving constraints

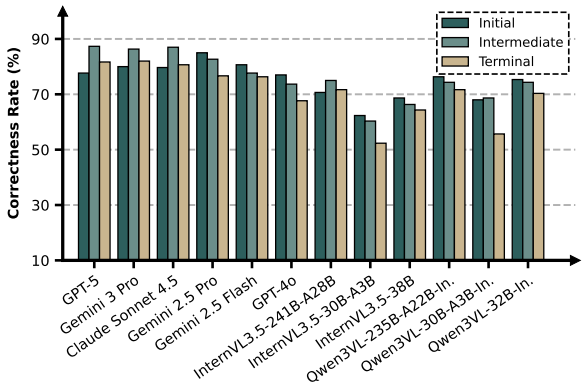

Figure 6: Step-wise Planning performance across different trajectory stages.

that may only become relevant later. Current models are noticeably stronger at the former than the latter, which explains why a single success rate from an execution benchmark can hide qualitatively different planning weaknesses.

5.3 Impact of Inference-Time Refinement

> ***Takeaway 2.*** Refinement substantially improves holistic planning, while its gains are smaller and less stable for feedback-conditioned step-wise decisions.

Recent advancements in test-time scaling have demonstrated that increasing inference-time compute can significantly bolster agentic performance (Gao et al., 2025; Yang et al., 2025b). To investigate the transferability of these benefits to planning, we evaluated self-refine and critic-guided strategies across four experimental settings (see Appendix C.8); the results are detailed in Table 4. In holistic planning, these methods deliver substantial gains, and critic-guided refinement with our error taxonomy is consistently strongest. For example, InternVL3.5-241B-A28B improves from 22.00% to 60.00% under taxonomy-guided critic refinement. This supports our central claim that holistic planning benefits from explicit diagnosis and revision. By contrast, step-wise planning already conditions on execution feedback and often concerns short-horizon decisions; adding extra reflection can induce over-correction or hallucinated alternatives (Appendix F.5). The executable validation in Table 3 further shows that improving holistic plans through APB-guided refinement can translate into better downstream execution metrics in controlled environments.

Benchmark	Model	Direct Exec.	Plan-first Exec.	Refine Exec.	APB CR	APB Grade
ToolSandbox	GPT-4o	76.3	79.2	**83.3**	61.5 → 84.5	0.66 → 0.87
ToolSandbox	Qwen3-VL-235B	80.7	79.1	**84.6**	75.5 → 88.0	0.79 → 0.92
ToolSandbox	Gemini 2.5 Flash	85.5	86.0	**89.4**	81.5 → 92.0	0.83 → 0.95
τ^2-bench	GPT-4o	0.54	0.56	**0.63**	35.0 → 65.5	0.54 → 0.78
τ^2-bench	Qwen3-VL-235B	0.47	0.58	**0.61**	32.5 → 62.5	0.45 → 0.79
τ^2-bench	Gemini 2.5 Flash	0.57	0.62	**0.69**	43.0 → 83.5	0.57 → 0.86

Table 3: **Executable validation.** Direct, Plan-first, and Refine report executable performance. APB CR and APB Grade report planning quality before and after APB-guided refinement ($a \rightarrow b$). ToolSandbox uses average trajectory similarity (%), and τ^2-bench uses average reward.

Method	Holistic CR (%)			Step-wise CR (%)			Holistic CR (%)			Step-wise CR (%)		
	Base	Refined	Δ	Base	Refined	Δ	Base	Refined	Δ	Base	Refined	Δ
	InternVL3.5-241B-A28B						**Qwen3VL-235B-A22B-Instruct**					
Self_R+	22.00	51.33	**+29.33**	79.33	72.00	**-7.33**	29.33	58.00	**+28.67**	78.00	74.00	-4.00
Critic_R+	22.00	60.00	**+38.00**	79.33	81.33	+2.00	29.33	62.00	**+32.67**	78.00	83.33	**+5.33**
Self_R-	22.00	28.67	**+6.67**	79.33	72.67	**-6.67**	29.33	35.33	**+6.00**	78.00	67.33	**-10.67**
Critic_R-	22.00	42.67	**+20.67**	79.33	82.00	+2.67	29.33	43.33	**+14.00**	78.00	82.67	+4.67
	InternVL3.5-30B-A3B						**Qwen3VL-30B-A3B-Instruct**					
Self_R+	9.33	22.00	**+12.67**	60.00	43.33	**-16.67**	17.33	38.00	**+20.67**	68.67	67.33	-1.33
Critic_R+	9.33	42.00	**+32.67**	60.00	68.00	**+8.00**	17.33	48.00	**+30.67**	68.67	78.00	**+9.33**
Self_R-	9.33	15.33	**+6.00**	60.00	51.33	**-8.67**	17.33	21.33	+4.00	68.67	57.33	**-11.33**
Critic_R-	9.33	36.00	**+26.67**	60.00	68.00	**+8.00**	17.33	34.67	**+17.33**	68.67	72.00	+3.33
	InternVL3.5-38B						**Qwen3VL-32B-Instruct**					
Self_R+	28.00	46.67	**+18.67**	67.33	66.67	-0.67	31.33	51.33	**+20.00**	74.00	74.00	+0.00
Critic_R+	28.00	53.33	**+25.33**	67.33	70.00	+2.67	31.33	56.67	**+25.33**	74.00	74.67	+0.67
Self_R-	28.00	30.67	+2.67	67.33	68.00	+0.67	31.33	26.67	-4.67	74.00	63.33	**-10.67**
Critic_R-	28.00	31.33	+3.33	67.33	70.67	+3.33	31.33	38.00	**+6.67**	74.00	74.00	+0.00

Table 4: **Impact of inference-time refinement.** Self_R and Critic_R compare Base and Refined CR; (+) uses error-taxonomy guidance.

Model	CR	Grade	Average Cost		
			$\mathcal{C}_{all}$	$\mathcal{C}_{pass}$	$\mathcal{C}_{com}$
GPT-5	89.5	**0.965**	13.21	13.35	13.55
Gemini 3 Pro	**91.2**	0.930	**12.54**	**12.65**	**12.94**
Gemini 2.5 Pro	89.5	0.958	13.70	13.86	13.73
Gemini 2.5 Flash	82.5	0.930	14.40	13.98	14.00
Claude Sonnet 4.5	82.5	0.937	15.40	16.02	16.15

Table 5: **Planning efficiency.** CR, Plan Grade, and average cost on all, solved, and commonly solved instances.

5.4 Efficiency-Aware Planning

> ***Takeaway 3.*** Top-tier models exhibit emerging economic rationality, simultaneously optimizing solution correctness and execution cost.

Operational efficiency is paramount for the scalable deployment of agents, a concern that has increasingly become the focus of recent research (Qu et al., 2025). To evaluate this capability, we curated a subset of 106 holistic planning instances where all five evaluated models originally achieved success. We then transformed these instances into a multi-solution environment by introducing alternative tools, assigning distinct execution costs to each. This setup shifts the objective from merely constructing a valid plan to identifying the optimal path that minimizes total cost while maintaining correctness.

As detailed in Table 5, results indicate a divergence in capability. Gemini 3 Pro demonstrates superior competence, maintaining the highest correctness rate (>91%) while achieving the lowest average cost. Conversely, Claude Sonnet 4.5 suffers significant performance degradation with markedly higher costs. These findings indicate that advanced models have evolved beyond merely identifying feasible solutions to exhibiting preliminary economic rationality, a critical step towards globally optimizing cost-effective agent systems.

6 Conclusion

We introduce APB, a planning-specific diagnostic benchmark for LLM agents. APB systematically evaluates holistic planning, feedback-conditioned step-wise planning, and robustness-oriented planning across 4,209 multimodal cases. Evaluations on 12 MLLMs confirm proprietary model superiority, yet reveal holistic planning's marked decline under extraneous tools. Regarding unsolvable scenarios, evaluations reveal that while top-tier models effectively detect explicit contradictions, all models remain vulnerable to implicit information gaps. We observe performance degradation in extended horizons and short-term reflection, alongside emerging economic rationality in proprietary models optimizing execution costs. We further validate APB's practical relevance through executable experiments on ToolSandbox and τ^2-bench. Together, these findings position APB as a foundational diagnostic testbed for diagnosing planning failures and advancing logically robust agentic systems.

Limitations

Despite our efforts to build a rigorous and scalable benchmark, we acknowledge that our work still has several limitations, summarized below:

- Although APB includes 4,209 multimodal test cases across 22 domains, it cannot exhaust the open-ended diversity of real-world agent tasks, and may under-represent highly specialized vertical areas (e.g., biomedicine or enterprise software).
- APB focuses on the planning component of agent behavior. Although our step-wise tasks include trajectory feedback and our executable validation demonstrates downstream utility in two controlled environments, APB should be used as a diagnostic complement to full end-to-end deployment benchmarks rather than as a replacement for them.
- APB is not a symbolic PDDL benchmark or a probabilistic MDP/POMDP benchmark. It evaluates plan synthesis from natural-language or multimodal inputs and fixed tool specifications, and therefore does not model stochastic transitions, belief-state updates, or action-outcome likelihoods.
- Parts of our pipeline rely on proprietary foundation models for judging and data synthesis. Although our calibration identifies Qwen3.5-397B-A17B as a strong self-hosted judge, deploying such a large model still requires substantial local compute, and the primary proprietary evaluator remains subject to provider-side updates.

Ethical considerations

Potential Risks This work proposes a benchmark for evaluating planning capabilities of LLM-based agents. A potential risk is that the benchmark may be over-optimized as a leaderboard target, leading to limited generalization beyond the benchmark. In addition, automatic evaluation using LLM judges may introduce biases or occasional misjudgments.

Personally Identifying Information and Offensive Content The dataset used in this work consists of synthetic tasks and environments, as well as annotations generated or curated by the authors and large language models. In addition, the benchmark incorporates selected data from existing datasets and a set of collected interaction records. As these sources may potentially contain personally identifying information, we conducted careful filtering and manual inspection to remove any information that could name or uniquely identify individuals, as well as offensive or inappropriate content.

Instructions Given to Participants This work did not involve external human participants, crowdworkers, or recruited annotators. All manual annotations were performed by the authors. Therefore, no participant instructions, consent forms, or risk disclosures were required.

References

Anthropic. 2025. Introducing claude sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. 2025. Tau2-bench: Evaluating conversational agents in a dual-control environment. arXiv preprint arXiv:2506.07982.

Rogerio Bonatti, Dan Zhao, Francesco Bonacci, Dillon Dupont, Sara Abdali, Yinheng Li, Yadong Lu, Justin Wagle, Kazuhito Koishida, Arthur Bucker, Lawrence Keunho Jang, and Zheng Hui. 2025. Windows agent arena: Evaluating multi-modal OS agents at scale. In Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025, volume 267 of Proceedings of Machine Learning Research. PMLR / OpenReview.net.

Ma Chang, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. 2024. Agentboard: An analytical evaluation board of multi-turn llm agents. Advances in neural information processing systems, 37:74325–74362.

Gautier Dagan, Frank Keller, and Alex Lascarides. 2024. Plancraft: an evaluation dataset for planning with llm agents. arXiv preprint arXiv:2412.21033.

Google DeepMind. 2025. Gemini 3 pro best for complex tasks and bringing creative concepts to life. https://deepmind.google/models/gemini/pro/.

Shihan Deng, Weikai Xu, Hongda Sun, Wei Liu, Tao Tan, Jianfeng Liu, Ang Li, Jian Luan, Bin Wang, Rui Yan, and Shuo Shang. 2024. Mobile-bench: An evaluation benchmark for llm-based mobile agents. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024, pages 8813–8831. Association for Computational Linguistics.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. Advances in Neural Information Processing Systems, 36:28091–28114.

Zane Durante, Qiuyuan Huang, Naoki Wake, Ran Gong, Jae Sung Park, Bidipta Sarkar, Rohan Taori, Yusuke Noda, Demetri Terzopoulos, Yejin Choi, Katsushi Ikeuchi, Hoi Vo, Li Fei-Fei, and Jianfeng Gao. 2024. Agent AI: surveying the horizons of multimodal interaction. CoRR, abs/2401.03568.

Pengfei Gao, Zhao Tian, Xiangxin Meng, Xinchen Wang, Ruida Hu, Yuanan Xiao, Yizhou Liu, Zhao Zhang, Junjie Chen, Cuiyun Gao, Yun Lin, Yingfei Xiong, Chao Peng, and Xia Liu. 2025. Trae agent: An llm-based agent for software engineering with test-time scaling. CoRR, abs/2507.23370.

Zefeng He, Xiaoye Qu, Yafu Li, Siyuan Huang, Daizong Liu, and Yu Cheng. 2025. Framethinker: Learning to think with long videos via multi-turn frame spotlighting. arXiv preprint arXiv:2509.24304.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2024. Metagpt: Meta programming for A multi-agent collaborative framework. In The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024. OpenReview.net.

Yue Huang, Jiawen Shi, Yuan Li, Chenrui Fan, Siyuan Wu, Qihui Zhang, Yixin Liu, Pan Zhou, Yao Wan, Neil Zhenqiang Gong, and Lichao Sun. 2024. Metatool benchmark for large language models: Deciding whether to use tools and which to use. In The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024. OpenReview.net.

Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W Mahoney, Kurt Keutzer, and Amir Gholami. 2024. An llm compiler for parallel function calling. In Forty-first International Conference on Machine Learning.

Haoming Li, Zhaoliang Chen, Jonathan Zhang, and Fei Liu. 2025. Planet: A collection of benchmarks for evaluating llms' planning capabilities. arXiv preprint arXiv:2504.14773.

Yuanchun Li, Hao Wen, Weijun Wang, Xiangyu Li, Yizhen Yuan, Guohong Liu, Jiacheng Liu, Wenxing Xu, Xiang Wang, Yi Sun, Rui Kong, Yile Wang, Hanfei Geng, Jian Luan, Xuefeng Jin, Zilong Ye, Guanjing Xiong, Fan Zhang, Xiang Li, and 6 others. 2024. Personal LLM agents: Insights and survey about the capability, efficiency and security. CoRR, abs/2401.05459.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, and 3 others. 2024. Agentbench: Evaluating llms as agents. In The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024. OpenReview.net.

Xiao Liu, Tianjie Zhang, Yu Gu, Iat Long Iong, Xixuan Song, Yifan Xu, Shudan Zhang, Hanyu Lai, Jiadai Sun, Xinyue Yang, Yu Yang, Zehan Qi, Shuntian Yao, Xueqiao Sun, Siyi Cheng, Qinkai Zheng, Hao Yu, Hanchen Zhang, Wenyi Hong, and 9 others. 2025. Visualagentbench: Towards large multimodal models as visual foundation agents. In The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025. OpenReview.net.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Haoping Bai, Shuang Ma, Shen Ma, Mengyu Li, Guoli Yin, Zirui Wang, and Ruoming Pang. 2025. Toolsandbox: A stateful, conversational, interactive evaluation benchmark for LLM tool use capabilities. In Findings of the Association for Computational Linguistics: NAACL 2025, Albuquerque, New Mexico, USA, April 29 - May 4, 2025, volume NAACL 2025 of Findings of ACL, pages 1160–1183. Association for Computational Linguistics.

Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, Rongcheng Tu, Xiao Luo, Wei Ju, Zhiping Xiao, Yifan Wang, Meng Xiao, Chenwu Liu, Jingyang Yuan, Shichang Zhang, and 7 others. 2025. Large language model agent: A survey on methodology, applications and challenges. CoRR, abs/2503.21460.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2024. Gaia: a benchmark for general ai assistants. In International Conference on Learning Representations, volume 2024, pages 9025–9049.

OpenAI. 2024. Hello gpt-4o. https://openai.com/index/hello-gpt-4o/.

OpenAI. 2025. Introducing gpt-5. https://openai.com/index/introducing-gpt-5/.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. Toolllm: Facilitating large language models to master 16000+ real-world apis. In The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024. OpenReview.net.

Xiaoye Qu, Yafu Li, Zhaochen Su, Weigao Sun, Jianhao Yan, Dongrui Liu, Ganqu Cui, Daizong Liu, Shuxian Liang, Junxian He, Peng Li, Wei Wei, Jing Shao, Chaochao Lu, Yue Zhang, Xian-Sheng Hua, Bowen Zhou, and Yu Cheng. 2025. A survey of efficient reasoning for large reasoning models: Language, multimodality, and beyond. CoRR, abs/2503.21614.

Qwen Team. 2025. Qwen3-vl technical report. CoRR, abs/2511.21631.

Yongliang Shen, Kaitao Song, Xu Tan, Wenqi Zhang, Kan Ren, Siyu Yuan, Weiming Lu, Dongsheng Li, and Yueting Zhuang. 2024. Taskbench: Benchmarking large language models for task automation. Advances in Neural Information Processing Systems, 37:4540–4574.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. Advances in Neural Information Processing Systems, 36:8634–8652.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2020. Alfworld: Aligning text and embodied environments for interactive learning. arXiv preprint arXiv:2010.03768.

Chenglei Si, Yanzhe Zhang, Ryan Li, Zhengyuan Yang, Ruibo Liu, and Diyi Yang. 2025. Design2code: Benchmarking multimodal code generation for automated front-end engineering. In Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 3956–3974.

Gemini Team. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. CoRR, abs/2507.06261.

Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2023. Planbench: An extensible benchmark for evaluating large language models on planning and reasoning about change. Advances in Neural Information Processing Systems, 36:38975–38987.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023a. Voyager: An open-ended embodied agent with large language models. arXiv preprint arXiv:2305.16291.

Jize Wang, Ma Zerun, Yining Li, Songyang Zhang, Cailian Chen, Kai Chen, and Xinyi Le. 2024. Gta: a benchmark for general tool agents. Advances in Neural Information Processing Systems, 37:75749–75790.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, Zhaokai Wang, Zhe Chen, Hongjie Zhang, Ganlin Yang, Haomin Wang, Qi Wei, Jinhui Yin, Wenhao Li, Erfei Cui, and 56 others. 2025a. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. CoRR, abs/2508.18265.

Xiaoxuan Wang, Ziniu Hu, Pan Lu, Yanqiao Zhu, Jieyu Zhang, Satyen Subramaniam, Arjun R Loomba, Shichang Zhang, Yizhou Sun, and Wei Wang. 2023b. Scibench: Evaluating college-level scientific problem-solving abilities of large language models. arXiv preprint arXiv:2307.10635.

Xinyuan Wang, Bowen Wang, Dunjie Lu, Junlin Yang, Tianbao Xie, Junli Wang, Jiaqi Deng, Xiaole Guo, Yiheng Xu, Chen Henry Wu, Zhennan Shen, Zhuokai Li, Ryan Li, Xiaochuan Li, Junda Chen, Boyuan Zheng, Peihang Li, Fangyu Lei, Ruisheng Cao, and 15 others. 2025b. Opencua: Open foundations for computer-use agents. In Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025.

Hui Wei, Zihao Zhang, Shenghua He, Tian Xia, Shijia Pan, and Fei Liu. 2025. Plangenllms: A modern survey of llm planning capabilities. arXiv preprint arXiv:2502.11221.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, and 9 others. 2025. The rise and potential of large language model based agents: a survey. Sci. China Inf. Sci., 68(2).

Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and

Yu Su. 2024a. Travelplanner: A benchmark for real-world planning with language agents. arXiv preprint arXiv:2402.01622.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. 2024b. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024.

John Yang, Carlos E Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. 2024. Swe-agent: Agent-computer interfaces enable automated software engineering. Advances in Neural Information Processing Systems, 37:50528–50652.

John Yang, Carlos E. Jimenez, Alex L. Zhang, Kilian Lieret, Joyce Yang, Xindi Wu, Ori Press, Niklas Muennighoff, Gabriel Synnaeve, Karthik R. Narasimhan, Diyi Yang, Sida Wang, and Ofir Press. 2025a. Swe-bench multimodal: Do AI systems generalize to visual software domains? In The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025. OpenReview.net.

Yan Yang, Dongxu Li, Yutong Dai, Yuhao Yang, Ziyang Luo, Zirui Zhao, Zhiyuan Hu, Junzhe Huang, Amrita Saha, Zeyuan Chen, Ran Xu, Liyuan Pan, Caiming Xiong, and Junnan Li. 2025b. GTA1: GUI test-time scaling agent. CoRR, abs/2507.05791.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. arXiv preprint arXiv:2210.03629.

Junjie Ye, Guanyu Li, Songyang Gao, Caishuang Huang, Yilong Wu, Sixian Li, Xiaoran Fan, Shihan Dou, Tao Ji, Qi Zhang, Tao Gui, and Xuanjing Huang. 2025. Tooleyes: Fine-grained evaluation for tool learning capabilities of large language models in real-world scenarios. In Proceedings of the 31st International Conference on Computational Linguistics, COLING 2025, Abu Dhabi, UAE, January 19-24, 2025, pages 156–187. Association for Computational Linguistics.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024. Webarena: A realistic web environment for building autonomous agents. In The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024. OpenReview.net.

Appendix Overview

This appendix presents supplementary materials organized into eight sections:

- **Appendix A: Comparison with Existing Benchmarks**. Compares APB with existing agent benchmarks across key dimensions.
- **Appendix B: Data Construction and Filtering Process**. Details the data augmentation pipeline, multi-stage quality control (rule-based, LLM-based, human verification), and the construction of diverse planning scenarios including holistic, robustness (tool-broken, tool-extraneous, unsolvable), and efficiency-aware tasks.
- **Appendix C: Evaluation Framework**. Outlines the comprehensive evaluation methodology, covering step-wise planning (one to three steps) with standardized error definitions, holistic planning emphasizing solvability, robustness settings (tool-broken, tool-extraneous, unsolvable, refinement), and efficiency-aware planning evaluation.
- **Appendix D: LLM-as-a-Judge Analysis**. Validates the automated evaluation pipeline through rigorous analysis of judge model selection, robustness to solution diversity, and self-evaluation bias.
- **Appendix E: Dataset Composition**. Describes the diverse dataset sources (FrameThinker, GAIA, GTA, OpenCUA, ToolBench, Real-World Interactions), presenting augmented query examples, step count statistics, and a detailed scenario taxonomy.
- **Appendix F: Data Samples and Model Response Examples**. Showcases qualitative examples of model performance across all settings, including correct/incorrect predictions, failure recovery patterns, extraneous tool impact, refinement effects, and responses to unsolvable constraints.
- **Appendix G: Detailed Experimental Results**. Presents granular quantitative results, including performance breakdowns by step range, modality, and planning stage, along with in-depth analysis of tool-extraneous impact and error distributions.
- **Appendix H: Model Sources and Hyperparameters**. Summarizes the sources, access methods (API vs. local), and hyperparameter configurations for all evaluated models.

Benchmark	Exec-Free	Process-Aware	Error Diag.	Step-wise	Holistic	Adversarial	Multimodal
AgentBench (Liu et al., 2024)	✗	✗	✗	✓	✗	✗	✗
AgentBoard (Chang et al., 2024)	✗	✓	✗	✓	✗	✗	✗
TaskBench (Shen et al., 2024)	✓	✓	✗	✗	✓	✗	✗
ToolEyes (Ye et al., 2025)	✗	✓	✓	✓	✗	✗	✗
MetaTool (Huang et al., 2024)	✗	✗	✓	✓	✗	✗	✗
GTA (Wang et al., 2024)	✗	✗	✗	✓	✓	✗	✓
PlanBench (Valmeekam et al., 2023)	✓	✗	✗	✗	✓	✗	✗
PlanCraft (Dagan et al., 2024)	✗	✓	✗	✓	✗	✓	✓
TravelPlanner (Xie et al., 2024a)	✗	✗	✗	✗	✓	✗	✗
VisualAgentBench (Liu et al., 2025)	✗	✓	✗	✓	✗	✗	✓
OSWorld (Xie et al., 2024b)	✗	✗	✗	✓	✗	✗	✓
WebArena (Zhou et al., 2024)	✗	✗	✗	✓	✗	✗	✓
Mind2Web (Deng et al., 2023)	✓	✗	✗	✓	✗	✗	✓
APB (Ours)	✓	✓	✓	✓	✓	✓	✓

Table 6: **Comparison of APB with existing agent benchmarks across key evaluation dimensions. Exec-Free**: Decouples planning from execution environment to avoid noise interference. **Process-Aware**: Evaluates intermediate reasoning processes beyond final outcomes. **Error Diag.**: Provides fine-grained error taxonomy for failure analysis. **Step-wise**: Supports local, short-horizon planning evaluation. **Holistic**: Supports global, long-horizon trajectory generation. **Adversarial**: Includes robustness scenarios (tool-extraneous, tool-broken, unsolvable tasks). **Multimodal**: Supports text, image, and video modalities.

A Comparison of APB with existing agent benchmarks

This section compares APB with existing agent benchmarks. Table 6 summarizes four primary categories of benchmarks contrasted with APB:

- **Execution-dependent benchmarks** (Liu et al., 2024; Ye et al., 2025; Huang et al., 2024) couple planning with execution, requiring interaction with real environments (e.g., web browsers, APIs). This introduces environmental stochasticity (e.g., network latency, API failures), masking true planning capabilities. In contrast, APB decouples planning from execution to assess pure planning logic without environmental interference.
- **Outcome-oriented benchmarks** (Wang et al., 2024; Xie et al., 2024a,b) evaluate performance primarily via final success rates, treating planning as a black box. This neglects intermediate reasoning and failure analysis. APB instead adopts a process-aware diagnostic paradigm, evaluating each planning step for fine-grained error attribution.
- **Holistic-only benchmarks** (Valmeekam et al., 2023; Xie et al., 2024a; Shen et al., 2024) require agents to generate complete plans in a single pass, assessing only global plan quality through structural metrics (e.g., graph similarity). While useful for end-to-end evaluation, these benchmarks lack step-wise assessment capabilities and fine-grained error analysis mechanisms. APB integrates both holistic and step-wise tasks, enabling multi-granularity evaluation with a comprehensive error taxonomy.
- **Reference-matching benchmarks** (Valmeekam et al., 2023; Shen et al., 2024; Deng et al., 2023) rely on exact or fuzzy matching against ground truth, assuming a single correct solution. APB employs a reference-aware LLM-as-Judge system to evaluate semantic correctness, recognizing diverse valid solutions.

B Data Construction and Filtering Process

B.1 Data Augmentation Pipeline

To evaluate agent planning in complex scenarios, we extend GAIA (Mialon et al., 2024), GTA (Wang et al., 2024), ToolBench (Qin et al., 2024), and FrameThinker (He et al., 2025) to increase query complexity, while retaining OpenCUA (Wang et al., 2025b) and real online traffic data as they are sufficiently representative of real-world complexity. We employ Claude Sonnet 4.5 to generate new queries Q from the original seed queries Q_{orig}, expand the corresponding tool sets $\mathcal{T}_{\text{orig}}$ to $\mathcal{T}$, and simulate the resulting execution trajectories τ. Post-augmentation, these datasets average 12.3 reasoning steps (an increase of 4.5) and 9.4 tools (an increase of 5.6) per query, significantly enhancing the difficulty and depth of the planning tasks.

Claude is therefore used for controlled augmentation rather than de novo benchmark generation. All six sources remain grounded in benchmark trajectories or real interaction records and subsequently pass the same validation pipeline. At evaluation time, records are normalized into the common APB schema. Tested models receive only task-facing fields—the system prompt, query, available tools, multimodal context, and, for step-wise tasks, a trajectory prefix. Generator identity, synthesis traces, and reference plans or continuations remain hidden. This separation prevents direct in-context copying of the generated reference and distinguishes source data, model-assisted augmentation, model-facing inputs, and evaluator-only supervision.

These controls do not establish the absence of pretraining contamination in proprietary models, which cannot be directly audited. We therefore restrict the evidence to observable checks. Claude Sonnet 4.5, the augmentation model, does not lead either core APB task: it obtains 64.1% Holistic CR and 77.3% Step-wise CR, compared with the respective best results of 74.5% and 80.1% in Table 1. In addition, APB-guided refinement improves native execution metrics in the independently constructed ToolSandbox and τ^2-bench environments (Table 3), providing an external transfer check beyond APB's construction pipeline.

B.2 Quality Control Pipeline

We implement a multi-stage quality control pipeline combining rule-based validation, model-based review, and human verification to ensure ground truth reliability.

- **Rule-based Structural Validation** Automated checks filter out non-existent tools and format violations (Figure 8).
- **LLM Task Reasonability Check** We verify query realism and tool validity, ensuring tools are not overpowered or structurally inconsistent (Figure 9).
- **LLM Trajectory Validation** Trajectories are evaluated using the **E1-E6 Error Taxonomy**. An LLM identifies errors in the query, tool list, and trajectory (Figure 10).
- **Human Verification** Finally, a three-author annotation team verifies consistency between objectives and logic, ensuring automated evaluation missed no flaws. Every one of the 1,109 final holistic base instances—including all 370 real-world instances—is independently reviewed by two members of this team. Each reviewer checks task solvability; tool definitions, schemas, arguments, and calls; consistency between tool returns and downstream dependencies; trajectory completeness and E1–E6 validity; multimodal grounding where applicable; and privacy and safety.

Disagreements between the two reviewers are adjudicated by the third member. When an instance requires manual correction, an author who did not perform that correction revalidates the complete corrected instance. Errors trigger iterative correction, and only data passing the automated checks, dual-author review, any necessary adjudication, and post-correction revalidation are retained as τ_{valid}. For real-world data, we additionally remove personally identifying information (PII) and sensitive content.

B.3 Step-wise to Holistic Transformation

We convert validated step-by-step trajectories into a holistic planning format to support diverse evaluation paradigms. This mitigates incompleteness from gradual information revelation in dialogues.

Using Claude Sonnet 4.5, we extract implicit information, rewriting Q_{orig} to Q_{refined} and reconstructing the plan P_{gt} and tool chain $\mathcal{C}_{\text{gt}}$. As detailed in the **Step-wise to Holistic Extraction Prompt** (Figure 11), this transformation prioritizes strategic reasoning. The process is formalized as:

$$(Q_{\text{refined}}, P_{\text{gt}}, \mathcal{C}_{\text{gt}}) = f_{\text{transform}}(Q_{\text{orig}}, \tau_{\text{valid}}) \quad (1)$$

where τ_{valid} is the validated trajectory. The generated $(P_{\text{gt}}, \mathcal{C}_{\text{gt}})$ undergo quality checks for validity, goal completeness, constraint satisfaction, and tool-chain consistency. They represent one validated feasible solution rather than a unique path or a certificate of global optimality.

B.4 Robustness Scenario Construction

We evaluate robustness via adversarial scenarios probing resilience to extraneous tools, failures, and unsolvable constraints.

B.4.1 Extraneous Tool Injection

We assess the agent's ability to select appropriate tools amidst distractors. The **Extraneous Tool Generation Prompt** (Figure 12) synthesizes semantically relevant but functionally inert tools, forcing precise semantic analysis.

Data Augmentation Prompt

You are an expert AI data augmentation and reasoning-chain generation specialist. Your mission is to take an existing simple dialogue (the seed query) and perform two subtasks:

TASK 1: Query Expansion & Toolset Enrichment
Transform the given simple user query into a **more complex, realistic, and analytically challenging query** — one that **naturally requires the introduction of new tools** to be solvable.
- The new query must:
  - Require multiple reasoning steps and multiple tool calls to complete.
  - Stay coherent, natural, and goal-driven (avoid artificial or random complexity).
  - Introduce interdependent subtasks when appropriate (e.g., data lookup → processing → comparison → conclusion).
  - Be sufficiently rich that solving it **demands capabilities beyond the current toolset**, thereby motivating the creation of new tools.
- You may invent **new tools** to support the expanded task, but:
  - Each invented tool must be **essential** for completing the new query (not optional or redundant).
  - Each invented tool must have a **clear purpose**, **realistic input/output**, and be **plausible** in an AI assistant environment.
  - New tools should fill well-defined capability gaps (e.g., new reasoning, analysis, or synthesis functions) that existing tools cannot address.
  - Avoid arbitrary or trivial tool creation; tools must emerge naturally from task needs.

TASK 2: Trajectory Generation and Tool Documentation
Using the new complex query and enriched toolset:
1. Simulate a complete reasoning and execution trajectory that solves the query.
2. Ensure each assistant step contains concise reasoning ("thought") and EXACTLY ONE tool call.
3. Include realistic tool outputs as tool steps.
4. End with a final assistant step calling the Finish function (or providing the final answer).
5. Collect all tools used (including pre-existing and newly created ones) and output them with complete, generic descriptions.

CRITICAL: Tool Calling Constraints
- Each assistant turn MUST call EXACTLY ONE tool (not zero, not multiple).
- ALL tool calls MUST succeed and return valid results — no failures, errors, or exceptions.
- Tool outputs must be realistic, specific, and directly usable by subsequent steps.
- Tools should be designed to be reliable and deterministic, producing successful results every time.

CRITICAL: Tool Description Requirements
- Tool descriptions must read like standalone documentation: clearly state the tool's purpose, enumerate every input parameter with name/type/defaults/-constraints, outline the returned structure, and include usage notes or best practices.
- Keep descriptions task-agnostic but specific enough that another engineer could confidently reuse the tool in unrelated contexts; avoid referencing the current task and never leak problem-specific data.
- Follow a polished textual style (e.g., "Does [X]. Parameters: 'param1' (string, required) - ... Returns ... Notes: ...") so each entry matches standard documentation.

AVAILABLE TOOLS (you can use these or invent new ones):
{List of Existing Tools}

ORIGINAL TASK CONTEXT:
{Original Question/Dialogue and Metadata}

OUTPUT FORMAT (CRITICAL)
You must output ONLY a valid single JSON object with this structure:

```
{
    "trajectory": [
        {"role": "user", "content": "expanded complex query"},
        {"role": "assistant", "thought": "brief reasoning", "tool_calls": [{"type": "function", "function": {"name": "
              function_name", "arguments": {}}}]},
        {"role": "tool", "name": "function_name", "content": "successful result with specific data"},
        ...
        {"role": "assistant", "tool_calls": [{"type": "function", "function": {"name": "Finish", "arguments": {"mode": "
              give_answer"}}}]}
    ],
    "tools": [
        {"tool_name": "ExistingTool1", "description": "..."},
        {"tool_name": "InventedToolA", "description": "..."}
    ]
}
```

REQUIREMENTS
- The output must be a single valid JSON object.
- Every assistant turn MUST have exactly ONE tool call.
- Every tool call MUST be followed by a successful tool response.
- Tool responses must contain realistic, specific, and usable data.
- Every tool used must be documented in the "tools" list.
- The process is irreversible — you cannot revise earlier steps.
- Target a specific number of reasoning steps (e.g., significantly more than the original).
- The new query must remain in the same domain but achieve higher complexity.
- The model should reason step-by-step WITHOUT knowing the final answer in advance.

Figure 7: Data augmentation prompt for transforming simple queries into complex, multi-step planning tasks with diverse tool requirements.

```python
def rule_based_verification(sample):
    """
    Stage 1: Structural and Logical Integrity Check
    """
    # 1. Format Validation
    required_fields = ["query", "trajectory", "tools
        "]
    for field in required_fields:
        if field not in sample:
            return False, f"Missing required field:
                {field}"

    # 2. Tool Existence Validation
    # Extract all defined tool names from the
        toolset
    available_tools = extract_tool_names(sample["
        tools"])

    # Extract all tool calls from the execution
        trajectory
    tool_calls = extract_tool_calls(sample["
        trajectory"])

    # Verify that every called tool exists in the
        definition
    for tool_name in tool_calls:
        if tool_name not in available_tools:
            return False, f"Hallucinated tool call
                detected: {tool_name}"

    return True, "Passed"
```

Figure 8: Pseudo-code for the rule-based verification stage, enforcing structural integrity and tool consistency in augmented data.

B.4.2 Tool-Broken Simulation

To evaluate adaptive planning, we simulate tool failures. The **Broken Tool Substitution Prompt** (Figure 13) generates functional replacements with distinct nomenclature, requiring the agent to identify and use the alternative.

B.4.3 Unsolvable Task Synthesis

We construct logically unsolvable tasks to test the agent planning models' capacity to refuse impossible requests, using four strategies:

- **Information Missing:** The **Information Missing Construction Prompt** (Figure 14) reformulates the query to depend on inaccessible private information.
- **Contradictory Constraints:** The **Contradictory Constraints Construction Prompt** (Figure 15) introduces constraints that invalidate standard methods.
- **Tool Removal:** The **Tool Removal Construction Prompt** (Figure 16) eliminates critical tools and mandates their use.
- **Visual Information Inaccessible:** The **Visual Information Inaccessible Construction Logic** (Figure 17) programmatically removes visual context to test detection of missing information.

These constructions instantiate a common missing-precondition principle: no valid plan exists when a necessary prerequisite is absent and cannot be recovered with the supplied context or tools. For example, a symbolic planning task that requires making tea while providing no cup has the same high-level failure structure as APB's missing-information, missing-visual-evidence, or removed-tool cases. APB represents such initial conditions through natural-language instructions, multimodal context, and tool availability rather than an explicit PDDL object inventory. The reported 400-instance evaluation remains restricted to the four implemented constructions above.

B.5 Efficiency-Aware Planning Construction

To evaluate cost-efficiency optimization, we construct the **Efficiency-Aware Planning** task. We augment holistic planning data with cost annotations and tools creating efficiency trade-offs. Each original tool receives a base cost (1 to 5), and we generate 5 to 10 new tools: (1) *Efficient Composite Tools* (Type A, 3 to 6 tools) combining 2 to 3 steps into one lower-cost tool, and (2) *Trap Tools* (Type B, 2 to 4 tools) that are suboptimal (overpriced, redundant, or off-target).

The goal is to ensure efficiency-aware agents use composite tools to minimize cost while avoiding trap tools. Figure 18 presents the generation prompt. All samples undergo manual verification.

B.6 Data Construction Case Studies

This section presents illustrative case studies covering data augmentation, holistic planning, robustness scenarios (including tool-broken, tool-extraneous, and unsolvable tasks), and efficiency-aware planning.

B.6.1 Data Augmentation Case Studies

We present case studies from GTA, GAIA, ToolBench, and FrameThinker to demonstrate the data augmentation pipeline's efficacy.

As shown in the following figures, the augmentation process consistently increases task complexity:

- **GTA (Figure 19):** Transforms a simple visual identification task into a multi-step financial workflow involving OCR, arithmetic reasoning, and policy compliance.
- **GAIA (Figure 20):** Expands a single-entity query into a comparative analysis across multiple journals, requiring weighted calculations and tabular synthesis.

Task Reasonability Check Prompt

Part 1: Query Reasonability Check
You are evaluating the reasonability of a task query for an AI agent benchmark.
IMPORTANT CONTEXT:
The AI agent being tested is a **multimodal agent** with full visual capabilities. It can:
• View and analyze images (photos, screenshots, diagrams, charts)
• Process video content
• Perform OCR (optical character recognition) on images
• Count objects in images
• Describe visual content
• Read text from product listings, menus, documents, etc.
Therefore, queries that involve visual tasks (analyzing photos, reading menus, counting objects in images, etc.) are all VALID and REASONABLE.
Task Query:
{query}
Evaluation Criteria:
1. **Realistic**: The query should represent a realistic task that a user might ask a multimodal AI agent
2. **Clear**: The query should be clear and understandable
3. **Achievable**: The task should be achievable with reasonable tools (search, calculator, file operations, image analysis, OCR, etc.)
4. **Not Overly Complex**: The query should not require superhuman capabilities
NOTE: Visual/image-related queries are completely valid since the agent has full visual capabilities.
Output Format (JSON only):

```
{
    "is_reasonable": true/false,
    "reason": "Brief explanation if not reasonable"
}
```

Evaluate and return your assessment.

Part 2: Tool Reasonability Check
You are evaluating the reasonability of AI-generated tools for a benchmark dataset.
IMPORTANT CONTEXT:
The AI agent being tested is a **multimodal agent** with full visual capabilities. It can:
• View and analyze images (photos, screenshots, diagrams, charts)
• Process video content
• Perform OCR (optical character recognition) on images
• Count objects in images
• Describe visual content
• Read text from product listings, menus, documents, etc.
Therefore, tools related to image processing, visual analysis, OCR, object detection, etc. are all VALID and REASONABLE.
Task Query:
{query}
Generated Tools to Evaluate:
{tools_json}
Evaluation Criteria:
1. **Realistic**: The tool should represent a realistic API/function that could exist in the real world
2. **Not Overpowered**: The tool should NOT be overpowered (e.g., "solve_any_problem", "get_perfect_answer")
3. **Appropriate Scope**: The tool's functionality should be appropriately scoped (not too broad, not too narrow)
4. **Consistent Format**: The tool definition should have proper structure (name, description, parameters)
NOTE: Image/vision-related tools (ImageDescription, OCR, CountGivenObject, etc.) are completely valid since the agent has visual capabilities.
Output Format (JSON only):

```
{
    "is_reasonable": true/false,
    "reason": "Brief explanation if not reasonable"
}
```

Evaluate ALL tools and return your assessment.

Figure 9: Task reasonability check prompt, validating both query realism and tool appropriateness for synthesized tasks.

Trajectory Logic Validation Prompt

You are evaluating the quality of an AI agent's trajectory for completing a task.
IMPORTANT CONTEXT:
The AI agent being tested is a **multimodal agent** with full visual capabilities.
[...details on multimodal capabilities...]

Therefore, tool calls related to image processing, visual analysis, OCR, object detection, etc. are all VALID and should NOT be flagged as errors.
Task Query:
{query}
Available Tools (first 10):
{tools_display}
Agent Trajectory:
{trajectory_display}
Error Classification Framework:
Evaluate the ENTIRE trajectory for the following error types:

1. **E1_GOAL_MISALIGNMENT**: Does any step deviate from or contradict the task goal?
2. **E2_PREMATURE_CONCLUSION**: Does the trajectory conclude before completing all required sub-tasks?
3. **E3_CONSTRAINT_VIOLATION**: Does any step violate explicit constraints from the query or format requirements?
4. **E4_LOGIC_ERROR**: Are there logical flaws, missing prerequisites, or illogical reasoning flows?
5. **E5_TOOL_USE_ERROR**: Are tools misused (wrong parameters, wrong data types, missing required parameters)?
6. **E6_HALLUCINATION**: Are there calls to non-existent tools or references to non-existent data?

IMPORTANT NOTES:
• Image/vision-related tools (ImageDescription, OCR, CountGivenObject, etc.) are completely valid since the agent has visual capabilities.
• The agent CAN see and process images, so using image analysis tools is NOT an error.
• Do NOT flag visual tool usage as hallucination or tool use error.

Output Format (JSON only):
```
{
    "is_valid": true/false,
    "errors": {
        "E1_GOAL_MISALIGNMENT": false,
        "E2_PREMATURE_CONCLUSION": false,
        "E3_CONSTRAINT_VIOLATION": false,
        "E4_LOGIC_ERROR": false,
        "E5_TOOL_USE_ERROR": false,
        "E6_HALLUCINATION": false
    },
    "error_details": "Detailed explanation of any errors found, or 'No errors' if valid"
}
```
Evaluate the trajectory and return your assessment:

Figure 10: Trajectory logic validation prompt with E1 to E6 error taxonomy for auditing coherence and validity of agent execution traces.

Step-wise to Holistic Extraction Prompt

You are an expert at reconstructing reasoning processes from execution trajectories. Your task is to extract a **first-person problem-solving plan** and the corresponding **tool usage chain** based on the given trajectory.

Task Objective

From the provided trajectory, extract:

1. **plan** — A concise, first-person summary describing how *I* planned to approach and solve the problem step by step.
2. **tool chain** — An ordered list of tools *I* used during the process, with each ‘reason‘ explaining *why I used that tool* in the problem-solving context.

Extraction Rules

- This is a **faithful extraction** task — do not rewrite or invent reasoning not present in the trajectory.
- Use **first-person narration** (e.g., "I first analyzed...", "I planned to use the search tool to find...") rather than meta references like "the model decided to..." or "the agent used...".
- The ‘plan‘ should describe the **planning and reasoning process only** — do NOT include actual results, outputs, or findings from tool executions.
- Focus on the **intended approach and strategy**, not on what was discovered or returned by the tools.
- The ‘plan‘ should be a coherent first-person narrative of the problem-solving strategy.
- The ‘tool_chain‘ should include:
  - ‘name‘: the exact tool name from the trajectory.
  - ‘reason‘: a brief, first-person explanation of *why I invoked this tool* or *what problem it solved* at that stage.
- Preserve the **original order** of tool invocations.
- Only include tools listed in the provided tool list.
- Output **only** a JSON object in this format:

```
{
    "plan": str,
    "tool_chain": [
        {"name": str, "reason": str},
        ...
    ]
}
```

Do not include any commentary or text outside the JSON object.

Task Query

{query}

Available Tools

{tools_str}

Trajectory to Analyze

{trajectory_str}

Figure 11: Step-wise to Holistic extraction prompt for distilling high-level planning strategies and tool chains from execution traces.

Extraneous Tool Generation Prompt

You are an expert in API design and testing. Your task is to generate "distractor" tools that will be mixed with real tools to test an AI agent’s ability to select the correct tool.

Context

- **User Query:** "{query}"
- **Current Trajectory:** {trajectory}... (truncated)
- **Existing Tools (Reference for Format and Context):**

{existing_tools_str}

Task

Please generate {count} distractor tools. All tools must be **Same-Scenario but Useless**:

- These tools must fit within the **EXACT SAME SCENARIO** as the user’s query but be **DEFINITELY USELESS** for the specific goal.
- They should look like they belong in the same API suite as the real tools (same naming convention, same resource type).
- **Example:** If the user wants to "book a flight", generate tools like ‘get_flight_seat_map‘, ‘check_airport_lounge_access‘, ‘download_airline_logo‘. These are about flights (same scenario) but cannot book a ticket (useless for goal).
- **Tool Description:** The description must be detailed and professional, using domain-specific terminology. It should reinforce that the tool is for the *same scenario* but performs a *different function* than what is needed.
- **Format Requirement:** You MUST strictly follow the JSON structure/schema of the **Existing Tools**. If existing tools use ‘tool_name‘ instead of ‘name‘, or have specific parameter formats, you MUST mimic them exactly.
- **CRITICAL:** The tool must be **IMPOSSIBLE** to use for the user’s specific goal. It should lack the specific functionality or parameters required to solve the current step of the problem.
- **CRITICAL:** Do NOT generate a tool that overlaps with the solution. It must be clearly distinct in function.
- **CRITICAL:** Do NOT generate tools that are completely unrelated. All tools must be relevant to the domain.

Requirements

- Return ONLY a JSON array of tool definitions.
- Ensure tool names are distinct from Existing Tools.
- Do NOT use generic names like ‘tool1‘, ‘useless_tool‘. Use descriptive, realistic names.

Figure 12: Extraneous tool generation prompt for synthesizing semantically relevant but functionally inert distractor tools.

Broken Tool Substitution Prompt

You are a helpful assistant that generates alternative tools for API development.
The tool '{tool_name}' is currently broken and cannot be used. Please generate a REPLACEMENT tool that serves the EXACT SAME purpose but has a COMPLETELY DIFFERENT name and slightly different description/parameters to avoid naming conflicts.
Original Tool Definition
{broken_tool_json}
Task Context
{context}
Requirements
1. The new tool MUST have a COMPLETELY DIFFERENT name.
   - DO NOT just add suffixes like '_v2', '_new', '_alt', '_1', etc.
   - DO NOT just add prefixes like 'new_', 'my_', etc.
   - Use synonyms or different phrasing (e.g., if original is 'find_hotels', use 'search_accommodation' or 'query_lodging').
2. The functionality MUST be identical (same inputs/outputs logic), but you SHOULD vary parameter names slightly if possible to make it look like a different library.
3. Return ONLY the JSON definition of the new tool.
4. If the original tool used "tool_name", the new tool MUST also use "tool_name". If it used "name", use "name".

Figure 13: Broken tool substitution prompt for generating functional replacements with distinct nomenclature to simulate tool failures.

Information Missing Construction Prompt

You are an expert at constructing adversarial test cases for **Multimodal AI Planners**. Your task is to modify a User Query to make it **Logically Unsolvable** due to **Missing Specific Information**.
The Logic Trap: You must rewrite the query so it refers to a specific parameter that the Agent **cannot know** and **cannot fetch** with the available tools.
Input Context:
- **Original Query:** "[Original Query]"
- **Original Solution Steps:** [Tool Chain]
- **Available Tools:** [Tool Bank]

CRITICAL CRE-CHECKS (You are blind to the image):
- **Tool Check:** Look at the `Available Tools`. If you introduce a "verification code" constraint, make sure there is NO tool like `ReadSMS`, `CheckEmail`, or `Get2FA` that could solve it. If such a tool exists, pick a different constraint (e.g., a verbal password).
- **Visual Singularity Risk:** Do NOT rely on adjectives like "the *red* car belonging to Bob". If there is only one red car in the image, the Agent will just select it.
  - *Bad:* "Select the **red** car that Bob likes." (If only 1 red car, Agent acts).
  - *Good:* "Select the car that matches the **license plate number I texted you**." (Invisible attribute, impossible to guess).

Strategies for "Hard Refusal" (Pick the best fit):
Strategy A: The "Private Reference" (Best for Search/Nav) Refer to information shared in a non-existent, external context (calls, meetings, physical notes).
- *Original:* "Search for **'Pizza Hut'**."
- *Strong Refusal:* "Search for the specific restaurant **we discussed in the morning meeting**." (Agent has no meeting logs).

Strategy B: The "Specific Input Requirement" (Best for Forms/Typing) Require a specific string or number that must be typed, which cannot be inferred visually.
- *Original:* "Rename the file to **'Final_Report'**."
- *Strong Refusal:* "Rename the file using the **standardized naming convention code** listed in the employee handbook." (Agent doesn't have the handbook).

Strategy C: The "Invisible Criteria" (Best for Selection) Make the selection depend on a hidden attribute, not a visual one.
- *Original:* "Click the **cheapest** item." (Price is visible, this is solvable).
- *Strong Refusal:* "Click the item that **is eligible for the 'Employee Friends & Family' discount**." (Eligibility is invisible).

Rules:
- **Keep the Visual Context:** Do not remove visual descriptions (e.g., "red button", "top menu") if they are part of the original intent, BUT add a logical gate that blocks the final action.
- **No Negatives:** Do not say "do not use tools".
- **Natural Tone:** The user should sound like they *forgot* the Agent doesn't know this info.

Output: Return **ONLY** the rewritten query string.

Figure 14: Information missing construction prompt for reformulating queries to depend on inaccessible private information.

Contradictory Constraints Construction Prompt

You are an expert at constructing adversarial test cases for **Multimodal AI Planners**. Your goal is to modify a User Query to make it **Logically Unsolvable**, forcing the agent to admit it cannot create a valid plan.
The Logic Trap: The Agent uses a powerful VLM (Visual Language Model). If you just say "The tool is down", the Agent might say "I can see the image, I will analyze it myself." **YOU MUST PREVENT THIS.** Your constraint must imply that **visual inspection alone is insufficient, misleading, or illegal** without the tool's specific metadata.
Input Context:
- **Original Query:** "[Original Query]"
- **Original Solution Steps:** [Tool Chain]
- **Available Tools:** [Tool Bank]

Constraint Strategies (Pick one that fits best):
Strategy A: The "Missing Dimension" (Best for Video/Gaia/Analysis) The visual data exists, but it lacks a critical invisible dimension required for the answer.
- *Bad:* "The face analysis tool is down." (Agent will just look at the face).
- *Good:* "Determine the emotion using the **biometric pulse data overlay** (currently corrupted). Visual expressions are misleading due to the actor's neurological condition and cannot be trusted."
- *Good:* "Analyze the chess position. Note that this is a variant where **piece colors are inverted relative to thermal signatures**, and only the `ThermalDecoder` (offline) can identify true ownership."

Strategy B: The "Compliance Lock" (Best for GTA/Counting/Finance) Make the *method* of data acquisition legally binding.
- *Example:* "For official audit, the count must be cryptographically signed by the `CountObject` tool. Unverified visual estimates are strictly prohibited by law."

Strategy C: The "Dynamic State" (Best for AgentNet/Web) The visual screenshot is outdated or static; reality requires a live check.
- *Example:* "Click the button, but only if the **server-side session token** (checked via `VerifySession`) is active. The session server is down."

Task Instructions:
- Identify the core dependency (e.g., Emotion, Count, Click).
- Create a constraint that makes the **Visual Information alone insufficient or untrustworthy**.
- **NEVER** suggest using "alternative methods" or "manual checks". Explicitly state that without the tool/data, the task **cannot be completed**.
- Append 1-2 natural sentences to the query.

Output: Return **ONLY** the modified query string (Original Query + Your Constraint).

Figure 15: Constraint conflict construction prompt for introducing constraints that invalidate standard operational methods.

Tool Removal Construction Prompt

You are an expert at constructing adversarial test cases for **Multimodal AI Planners**. The goal is to create an "Unsolvable" test case by **Removing Critical Capabilities** and **Enforcing Tool Dependency**.
The Logic: We want the Agent to realize: *"I need to perform this action using a tool to be compliant, but I have NO tools available to do it."*
Your Task 1: Select Tools to Remove (The "Capability Void")
- Identify the **core capability** required (e.g., Counting, Clicking, OCR, Searching).
- Remove **ALL tools** associated with that capability (primary AND backups) to ensure no alternative exists.
- **Target:** Action Tools (click, type) or specialized Perception Tools (Count, OCR, Detect).
- **Select AT LEAST 2 tools** if possible to ensure a complete block.

Your Task 2: Rewrite the Query (The "Compliance Constraint")
- Append a constraint to the User Query that **forbids** using internal capabilities (like "visual estimation" or "mental calculation").
- **CRITICAL:** Do **NOT** name the specific tools (e.g., don't say "Use CountObject"). Instead, use **generic functional descriptions**.
  - *Bad:* "Use `CountGivenObject` to count." (Too obvious contradiction).
  - *Good:* "For audit compliance, the count must be **generated by a verified counting tool**. Visual estimates are strictly prohibited."
  - *Good:* "You must use **programmatic navigation actions** to interact with the map. Manual interactions are not logged and therefore invalid."
- **NO** "IMPORTANT", "WARNING", "COMPLIANCE", "NOTE:", or bold headers.

Input Context:
- **Original Query:** "[Original Query]"
- **Ground Truth Tool Chain:** [Tool Chain]
- **Available Tools:** [Tool Bank]

Output Format: Return **ONLY** a JSON object:

```
{
"tools_to_remove": ["ToolName1", "ToolName2", ...],
"modified_query": "The new query string with the generic compliance constraint..."
}
```

Figure 16: Tool removal construction prompt for creating logical deadlocks by eliminating critical tools while mandating their use.

```python
def construct_visual_missing_case(sample):
    """
    Constructs a 'Visual Information Inaccessible' unsolvable case by stripping visual context.

    """
    # 1. Preserve the original query
    # We deliberately keep the query unchanged (e.g., "Describe the image")
    # to test if the agent can correctly identify that the image is missing.
    query = sample["query"]

    # 2. Remove Visual Information
    # The core logic is to empty the file list in the background context,
    # effectively simulating a system error where the image failed to load
    # or was not provided.
    if "background" in sample:
        sample["background"]["files"] = []

    # 3. Mark as Unsolvable
    sample["unsolvable_type"] = "visual_missing"

    return sample
```

Figure 17: Visual missing construction logic for programmatically removing visual data to test agent detection of missing context.

Efficiency-Aware Tool Bank Generation Prompt

System Prompt:
You are an expert in designing Agent Planning Benchmarks. Your goal is to REWRITE the 'tool_bank' for a given task to test 'Plan Efficiency'.

Core Logic:
You must provide a tool set where multiple paths exist to solve the problem, but one path (using Composite Tools) is significantly cheaper (lower total cost) than others.

Instructions:

- **Original Tools (Mandatory):** Keep ALL original tools. Assign them a reasonable base cost (e.g., integers between 1 and 5).
- **ADD 5 to 10 NEW Tools (Strict Constraint):** Generate exactly 5 to 10 new tools divided as follows:
  - **Type A: Efficient Composite Tools (3 to 6 tools):** Combine 2 to 3 consecutive steps from the Ground Truth into one single tool. The cost MUST be LOWER than the sum of the original tools it replaces (e.g., if Tool A cost 2 and Tool B cost 2, Composite Tool AB should cost 3).
  - **Type B: Inefficient/Trap Tools (2 to 4 tools):** Tools that look relevant but are suboptimal: *Overpriced* (simple job, very high cost), *Redundant Steps* (breaks simple action into unnecessary sub-steps), or *Slightly Off-target* (retrieves partial information, forcing more tool calls).
- **NO Magic Tools:** Do not create a single tool that solves the whole query in one step.
- **Output Format:** Return a SINGLE valid JSON list containing [All Original Tools + All New Tools]. Each tool object must have: `name`, `description`, `parameters`, and `cost` (integer).

User Message:
User Query: {query}
Original Ground Truth / Steps: {tool_chain or plan}
Original Tool Bank: {original_tools}
Requirement: Generate a JSON list of tools (original + composite + traps) with costs. Output PURE JSON only.

Figure 18: Efficiency-aware tool bank generation prompt for creating cost-annotated tools with composite (efficient) and trap (inefficient) alternatives.

- **ToolBench (Figure 21):** Evolves a weather check into an event planning scenario with contingency management and document generation.
- **FrameThinker (Figure 22):** Converts a multiple-choice visual question into a narrative analysis, demanding character and emotional tracking.

These transformations ensure the benchmark rigorously probes long-horizon planning and reasoning.

B.6.2 Holistic Planning Case Studies

Figure 23 illustrates the transformation from step-wise to holistic planning. We convert execution trajectories (with observations) into a holistic format consisting of a high-level **Plan** (synthesized strategy) and an observation-free **Tool Chain** (sequence of tools).

B.6.3 Robustness Case Studies

This section details Tool-Broken, Tool-Extraneous, and Unsolvable scenarios.

Figure 24 illustrates a Tool-Broken scenario. The agent is planning a romantic dinner and attempts to fetch air quality data using `FetchAirQualityAPI`, but the tool is simulated to be broken. The agent must identify this failure and switch to the designated replacement tool, `GetAtmosphericPollutionData`.

Figure 25 depicts a Tool-Extraneous scenario. The system injects several extraneous tools (e.g., `stock_v2_get_company_logo`) that appear related to the entity but are functionally irrelevant. The agent must ignore these distractors and use only the necessary tools.

Figure 26 presents examples for each of the four Unsolvable categories: Contradictory Constraints, Information Missing, Tool Removal, and Visual Information Inaccessible.

B.6.4 Efficiency-Aware Planning Case Studies

Figure 27 demonstrates Efficiency-Aware Planning. The tool bank includes original tools, *Efficient Composite Tools* (Type A) that combine multiple steps for cost reduction, and *Trap Tools* (Type B) that appear relevant but are suboptimal (overpriced or redundant).

C Evaluation Framework

C.1 Planning Assumptions and Scope

APB is an execution-free plan-synthesis benchmark over natural-language or multimodal queries, system instructions, and tool schemas. It is not formulated as PDDL, an MDP, or a POMDP: it does not define symbolic state variables, action preconditions and effects, belief states, transition distributions, or outcome probabilities. Holistic Planning assumes that the supplied tools function as specified and provides no intermediate feedback. Step-wise Planning conditions on a fixed observed trajectory prefix and its recorded tool returns. For augmented trajectories, tool calls are designed to be deterministic and successful; Tool-Broken Planning instead supplies one fixed observed failure and a designated alternative rather than sampling a stochastic transition. Information Missing and Visual Information Inaccessible evaluate feasibility and refusal, not probabilistic inference over latent states. Consequently, APB neither supplies action-outcome likelihoods nor evaluates a model's ability to estimate them.

C.2 Evaluation Metrics

We employ three complementary metrics: **Plan Correctness** (pass/fail), **Plan Grade** (0 to 1 scale), and **Error Taxonomy** (E1 to E6). The E1–E6 taxonomy was not manually imposed a priori; we first inspected representative planning failures, used LLMs to cluster and summarize recurring error patterns, and then finalized the categories through human checks for semantic independence and annotatability. To systematically diagnose failure mechanisms, the final taxonomy contains **six** semantically independent error categories, where a sample may exhibit multiple errors simultaneously. Notably, E2 instantiates two protocol-specific subtypes that capture a key distinction between step-wise and holistic planning.

C.2.1 Step-wise Planning Metrics

Step-wise evaluation enforces **zero tolerance**: any flaw constitutes an error. A step is correct only if it contains zero errors (E1 to E6 all False). The focus is on **correctness and reasonableness**, not optimality.

Plan Grade (Step-wise).

- **1.0 (Perfect):** No errors. The step is correct, logical, and progresses toward the goal. *Note: The step does NOT need to be "optimal"; it just needs to be correct and reasonable.*
- **0.8 (Very Good):** At most one very minor issue (E5 only with trivial impact) that does not affect correctness.

GTA Augmentation Example

Original Sample
Query: "How much should I pay for the beer on the table according to the price on the menu?"
Trajectory Summary (5 Steps):

- `ImageDescription`: Describe the beer bottles on the table.
- `ImageDescription`: Describe the menu image.
- `CountGivenObject`: Count the number of beer bottles.
- `OCR`: Read the beer brand from the bottle label.
- `OCR`: Read the price from the menu.

Augmented Sample
Query: "I'm dining at this beachfront restaurant and need a detailed expense breakdown. First, identify what beverages are on my table and what menu is shown. Then calculate the total cost for all beverages, apply the restaurant's 10% service charge and 7% sales tax, split the final bill among 3 people, verify the origin country of the beer brand, check if any promotional discounts apply for ordering multiple beers, and finally provide a complete itemized receipt with per-person cost."
Trajectory Summary (11 Steps):

- `ImageDescription`: Identify beverages (beer bottles) in the first image.
- `ImageDescription`: Confirm the second image is the menu.
- `CountGivenObject`: Count the exact number of beer bottles (2).
- `OCR`: Read the beer brand ("MAGNA") from the bottle.
- `OCR`: Read prices from the menu image.
- `ParseMenuItems`: Structure the menu data to find the price of Magna beer.
- `CheckPromotionalDiscount`: Check for quantity discounts (found 5% off).
- `VerifyProductOrigin`: Determine the origin of the beer (Puerto Rico).
- `CalculateBillWithCharges`: Calculate total with discount, service charge, and tax.
- `SplitBillAmongPeople`: Split the total amount among 3 people.
- `GenerateItemizedReceipt`: Generate a detailed receipt with all findings.

Figure 19: GTA augmentation example: transforming a simple visual identification task into a multi-step financial analysis workflow requiring OCR, arithmetic reasoning, and policy compliance.

GAIA Augmentation Example

Original Sample
Query: "If we assume all articles published by Nature in 2020 ... relied on statistical significance ... and they on average came to a p-value of 0.04, how many papers would be incorrect ...?"
Trajectory Summary (2 Steps):

- `search_engine`: Find the number of articles published in Nature in 2020.
- `calculator`: Calculate 4% of the total count.

Augmented Sample
Query: "I'm investigating the reliability of statistical claims in high-impact journals. For Nature in 2020, I need to: (1) determine how many research articles were published, (2) calculate how many would be false positives if all used p=0.04, (3) compare this false positive rate against three other major journals (Science, Cell, and The Lancet) ... (4) adjust each journal's false positive count by their respective impact factors ... (5) generate a ranked comparison table, and (6) calculate what the aggregate false positive rate would be ..."
Trajectory Summary (12 Steps):

- `search_engine` & `calculator`: Retrieve article counts and calculate false positives for *Nature*.
- `search_engine` & `calculator`: Repeat for *Science*.
- `search_engine` & `calculator`: Repeat for *Cell*.
- `search_engine` & `calculator`: Repeat for *The Lancet*.
- `search_engine`: Find 2020 impact factors for all four journals.
- `weighted_metric_calculator`: Calculate influence-weighted false positive counts.
- `tabular_data_formatter`: Create a ranked comparison table.
- `calculator`: Compute the aggregate false positive count.

Figure 20: GAIA augmentation example: expanding a single-entity statistical query into multi-journal comparative analysis with weighted calculations and tabular synthesis.

ToolBench Augmentation Example

Original Sample
Query: "I'm planning a romantic dinner on a rooftop terrace. Can you provide me with the sunset time for the date of the dinner? Additionally, I'd like to know the current weather conditions and a 5-day forecast ..."
Trajectory Summary (3 Steps):

- `get_sunrise_and_sunset_times`: Get sunset time.
- `weather_report`: Get current weather.
- `get_5_day_forecast`: Attempt to get forecast (failed in original).

Augmented Sample
Query: "I'm planning a romantic anniversary dinner ... I need comprehensive planning data: exact sunset time with twilight phases, current weather, detailed 5-day and 16-day forecasts, historical weather patterns ..., solar position ..., UV index, air quality, wind patterns at rooftop level ..., precipitation ..., temperature trends ... cross-reference with restaurant reservation availability, backup indoor venue ..., generate risk assessment ..., compile PDF report ..."
Trajectory Summary (14 Steps):

- `get_sunrise_and_sunset_times`: Get detailed sunset and twilight data.
- `weather_report`: Get current baseline conditions.
- `get_5_day_forecast` & `view_16_day_forecast`: Get short and long-term forecasts.
- `climate_forecast` & `QueryHistoricalWeather`: Analyze statistical weather patterns.
- `sunposition`: Calculate solar angle for table placement.
- `FetchAirQuality` & `QueryWindProfile`: Check comfort factors (AQI, rooftop wind).
- `QueryReservation`: Find available rooftop tables.
- `GenerateBackupVenueAnalysis`: Plan contingencies for bad weather.
- `GenerateWeatherRiskAssessment`: Synthesize all data into a risk report.
- `GeneratePDFReport`: Compile findings into a document.
- `ConfigureWeatherAlertSystem`: Set up active monitoring.

Figure 21: ToolBench augmentation example: transforming a simple weather query into comprehensive event planning with contingency management and document generation.

FrameThinker Augmentation Example

Original Sample
Query: "The scene where a large number of foxes surround the girl at 2:56 implies what? A: The fox clan collectively seeks revenge. B: The girl develops schizophrenia. C: The underground imprisonment will continue. D: Mom comes to the rescue. E: The fox is preparing to execute the girl. F: The girl gains superpowers."
Trajectory Summary (3 Steps):

- `get_frame_number`: Convert timestamp "2:56" to frame number (4224).
- `choose_frames`: Request frames around frame 4224 to analyze the scene.
- `output_answer`: Conclude "C: The underground imprisonment will continue" based on visual analysis.

Augmented Sample
Query: "I need you to perform a comprehensive narrative and thematic analysis of this stop-motion animated sequence. Please analyze the following aspects systematically: A) Identify and catalog all distinct environmental settings... B) Track the protagonist's emotional arc... C) Detect and analyze the presence and behavior of the fox/rabbit-like creatures... D) Synthesize your findings... to determine the thematic significance of the scene around frame 4224..."
Trajectory Summary (8 Steps):

- `classify_environment`: Catalog settings across all frames (underground vs. domestic).
- `detect_character_presence`: Track the girl's visibility across frames.
- `analyze_emotional_state`: Analyze the girl's emotions (fear, distress) in key frames.
- `detect_character_presence`: Track the fox/rabbit creatures' presence.
- `filter_frames`: Isolate frames around the target timestamp (4224).
- `analyze_spatial_relationship`: Analyze proximity and power dynamics between girl and creature.
- `analyze_narrative_theme`: Evaluate thematic evidence for captivity vs. other interpretations.
- `synthesize_thematic_significance`: Synthesize all findings to determine the scene's meaning (Ongoing Captivity).

Figure 22: FrameThinker augmentation example: converting a multiple-choice visual question into systematic multi-dimensional narrative and thematic analysis.

Holistic Data Construction Example

Original Step-wise Data (Input) *Contains the full interaction history with observations:*

- **Step 1:** Call `get_sunset_time(...)`
- *Observation:* "Sunset is at 20:12."
- **Step 2:** Call `get_weather(...)`
- *Observation:* "Clear skies, 18°C."
- ... (Full trajectory with 14 steps and observations)

↓ **Transformation Process** ↓

Transformed Holistic Data (Output) *Synthesized Plan and Extracted Tool Chain (No observations):*
1. Synthesized Plan: "I planned to approach this comprehensive dinner planning task by systematically gathering all required data in logical sequence. First, I would obtain the exact sunset time... Next, I would establish baseline weather conditions... then progressively build confidence through multiple forecast horizons... Finally, I would compile everything into a formatted PDF report..."
2. Extracted Tool Chain:

- `get_sunset_time`
- `weather_report`
- `get_5_day_forecast`
- `view_16_day_forecast`
- `calculate_solar_position`
- ... (Sequence of 14 tools)

Figure 23: Data transformation example: converting step-wise trajectory into descriptive **Plan** with extracted **Tool Chain** for holistic planning ground truth.

Tool-Broken Case Study

Trajectory Prefix:

- call: GetSunsetTime(date='2024-05-18', location='San Francisco')
- call: GetWeatherForecast(date='2024-05-18', location='San Francisco')
- ... (Previous successful steps)

Broken Tool: `FetchAirQualityAPI` (Simulated Failure)
Designated Replacement Tool: `GetAtmosphericPollutionData`

- Description: Obtains real-time and projected atmospheric pollution metrics including air quality index measurements and specific contaminant levels from environmental sensor networks. Parameters: 'lat', 'lon', 'target_date'. Returns JSON object containing present AQI measurement and health advisory information.

Figure 24: Tool-broken case study from ToolBench: agent recovery from `FetchAirQualityAPI` failure by utilizing the alternative `GetAtmosphericPollutionData`.

Tool-Extraneous Case Study

User Query: I need to conduct a comprehensive multi-dimensional analysis of stock performanceId 0P0000OQN8... Please provide: (1) Complete security information... (2) Key financial ratios... (3) Detailed key statistics... (5) Current order book depth...
Injected Extraneous Tools (Distractors):

- `stock_v2_get_company_logo_for_morning_star`: Retrieves the official company logo and branding assets. (Irrelevant to quantitative analysis)
- `stock_v2_get_executive_profiles_for_morning_star`: Fetches biographical information of executives. (Irrelevant to order book/price analysis)
- ...

Figure 25: Tool extraneous case study: agent must avoid semantically related but task-irrelevant distractor tools during financial analysis.

Unsolvable Case Studies

1. Contradictory Constraints: *Query:* "Open your store on the Steam Store homepage... filter by date... save as default preference... Note that due to GDCR compliance requirements, all preference-saving operations must be verified through the `SessionAuthValidator` tool... The `SessionAuthValidator` service is currently experiencing a critical outage..."
2. Missing Information: *Query:* "Update twitter profile: add the bio prefix that I mentioned in our phone call earlier before the current bio, and change the location to the city code from the company's regional office list." (The specific bio prefix and city code are not provided in the context).
3. Tool Removal: *Query:* "Please search for the nearest open pizza place... All interface interactions must be executed through registered interaction handlers..." *Available Tools:* `["moveto", "dragto", "write", "press", "hotkey", "scroll"]` (The critical `click` tool is removed).
4. Visual Information Inaccessible: *Query:* "Please search for the nearest open pizza place to my home on Google Maps..." *Visual Context:* `[]` (The necessary screenshot/visual interface is not provided).

Figure 26: Unsolvable scenario examples: tasks blocked by external constraints (service outage), missing information (unknown bio prefix), missing tools (no click action), or missing visual context (no screenshot).

Efficiency-Aware Planning Case Study

User Query: I need a comprehensive analysis of this supernatural ritual sequence. Using the following 16 frames, please identify the chronological progression of ritual phases, track emotional arc, and detect the manifestation moment...

Original Tools (Base Costs):
- `classify_sequence_phases`: cost=3
- `analyze_emotional_state`: cost=2
- `detect_text_appearance`: cost=2
- `compare_frames`: cost=2
- `classify_environment`: cost=3

Efficient Composite Tools (Type A: Lower Combined Cost).
- `analyze_ritual_progression_with_emotion`: cost=4 (combines phase classification + emotional analysis, saves 1 vs. separate calls)
- `detect_manifestation_moment`: cost=3 (combines text detection + frame comparison, saves 1)
- `comprehensive_ritual_analysis`: cost=6 (combines phase + emotional + environment analysis, saves 2)

Trap Tools (Type B: Overpriced or Redundant).
- `extract_individual_frame_metadata`: cost=8 (extracts detailed metadata for single frame, which is overpriced)
- `analyze_micro_expressions`: cost=7 (micro-expression analysis, which has redundant granularity)
- `detect_ambient_sound_correlation`: cost=5 (audio analysis, which is off-target for visual task)

Figure 27: Efficiency-aware planning case study: tool bank with original tools, cost-efficient composite alternatives, and overpriced trap tools.

- **0.6 (Acceptable):** One minor error (E4 or E5) that does not fundamentally break the solution.
- **0.4 (Problematic):** Multiple minor errors (2+ of E4/E5) or one moderate error (E2).
- **0.2 (Severe):** Major errors (E1, E3, or E6) or multiple moderate errors.
- **0.0 (Failed):** Multiple major errors or complete logical breakdown.

Error Taxonomy (Step-wise).

- **E1: Goal Understanding Error**. The step deviates from or contradicts the current goal based on the query and previous steps.
- **E2: Premature Conclusion**. The step attempts to conclude or output results when necessary intermediate steps are still missing (e.g., outputting results without calling necessary tools).
- **E3: Constraint Violation**. The step violates an explicit constraint from the query, previous steps, or the required output format (e.g., time range, data source, format, method).
- **E4: Logic Error**. The step has logical flaws or potential risks: requires data/results not yet obtained, ignores available data from previous steps, or has illogical reasoning flow.
- **E5: Tool Use Error**. The step misuses a tool's function or passes incorrect parameters according to the tool's specification (strict parameter check: vague or non-specific parameters are errors).
- **E6: Hallucination**. The step calls a non-existent tool or references non-existent data. Tool names must match *exactly*; even slight variations are errors.

C.2.2 Holistic Planning Metrics

Holistic evaluation prioritizes **Solvability** over **Similarity**, assessing if the plan logically solves the user's intent. Key principles:

- **Use the Expert as Reference for Logic:** The expert example shows one valid way to solve the problem. **If the model's approach is fundamentally different, critically analyze whether it is also logically viable.**
- **Detect Multiple Errors:** A plan often fails in multiple ways. Evaluate all 6 categories independently; do not stop after finding the first error.
- **Evaluate the Sequence:** Focus on the linear sequence of the tool chain, checking for correct data dependencies and logical omissions.

Plan Grade (Holistic).

- **1.0 (Correct):** All errors are 0. Plan is logically sound and solves the query.
- **0.8 (Very Good):** Almost perfect; follows expert logic closely with a minor, non-critical error. Core logic is 90% correct.
- **0.6 (Mostly Correct):** Captures the main logical flow but misses or flaws one key component: "on the right track" but fails.
- **0.4 (Partially Correct):** Identifies some correct steps but the overall sequence is wrong or fundamentally broken.
- **0.2 (Mostly Incorrect):** Deeply flawed; might get one simple step right but completely fails the core logic.
- **0.0 (Failed):** Zero logical merit; complete hallucination or goal misunderstanding.

Error Taxonomy (Holistic).

- **E1: Goal Understanding Error**. Fundamentally misunderstands the user query's core intent.
- **E2: Task Incompleteness**. Fails to plan for all required sub-tasks of a multi-part query.
- **E3: Constraint Violation**. The plan violates an explicit constraint from the user query or system prompt.
- **E4: Logic Error**. The logical reasoning doesn't match; steps lack key premises, have circular arguments, or make the query unsolvable.
- **E5: Tool Use Error**. The plan misunderstands a tool's function or its required data type.
- **E6: Hallucination**. Calls a non-existent tool, outputs factual errors contrary to common sense, or uses results not yet available.

C.3 Step-wise Planning Paradigm

The step-wise paradigm assesses immediate action prediction across three horizons:

- **One-step Planning:** Predict the single subsequent step (Figure 28).
- **Two-step Planning:** Predict the next two consecutive steps (Figure 29).
- **Three-step Planning:** Predict the next three consecutive steps (Figure 30).

Evaluation uses a judge-mediated protocol. The **Standardized Evaluation Template** (Figure 31) applies detailed **Error Definitions** (Figure 32) and a **Scoring Rubric** (Figure 33). Any detected error (E1-E6) results in failure.

C.4 Holistic Planning Paradigm

The holistic paradigm requires generating a complete, end-to-end plan before execution. This tests strategic decomposition without intermediate feedback. The **Holistic Planning Prediction Prompt** (Figure 34) guides generation. The **Holistic Planning Evaluation Prompt** (Figure 35) assesses logical flow, tool selection, and parameter correctness against ground truth.

C.5 Tool-Broken Planning Setting

This setting evaluates the agent's adaptability and resilience when confronting tool failures. The agent is presented with a scenario where a specific tool call has returned an error. The **Tool-Broken Evaluation Prompt** (Figure 36) classifies the agent's subsequent behavior into five distinct categories:

- **Replace:** Successfully utilized the designated replacement tool.
- **Alter:** Successfully employed another valid tool to solve the problem.
- **Retry:** Failed by attempting to re-invoke the broken tool.
- **Refusal:** Failed by explicitly abandoning the task.
- **Other:** Failed by employing irrelevant tools or exhibiting hallucinations.

C.6 Tool-Extraneous Planning Setting

This setting probes the agent's robustness against irrelevant information and distraction. We inject distractor tools into the available toolset.

For the **Step-wise Setting**, the **Step-wise Tool-Extraneous Prediction Prompt** (Figure 37) presents these mixed tools to the agent. For the **Holistic Setting**, the **Holistic Tool-Extraneous Prediction Prompt** (Figure 38) is used.

The evaluation is performed using the **Step-wise Tool-Extraneous Evaluation Prompt** (Figure 39) for the step-wise setting, and the **Holistic Tool-Extraneous Evaluation Prompt** (Figure 40) for the holistic setting. Both explicitly flag the utilization of any distractor tool as a critical error (E5_TOOL_USE_ERROR).

C.7 Unsolvable Planning Setting

In the Unsolvable Planning setting, we evaluate the agent's robustness against impossible tasks. The evaluation process follows a strict judge-based protocol:

- **Task Presentation:** The agent is presented with a query that is unsolvable due to missing visual input, removed tools, conflicting constraints, or missing information.
- **Response Generation:** The agent attempts to plan or refuses the task.
- **Judgement:** A Judge Model evaluates the agent's response using specific criteria for each unsolvable category (as shown in Figure 41). The judge checks for explicit refusal (Pass) versus hallucination or blind execution (Fail).

C.8 Inference-Time Refinement Setting

To evaluate the effectiveness of inference-time refinement, we introduce a refinement stage where the agent (or an external critic) reflects on the initial prediction before generating the final output. This process consists of three steps:

One-step Planning Inference Prompt

You are solving a task step by step only using the available tools.
Task: [User Query]
Available Tools:
- [Tool Name]: [Description] Parameters: [Params]
- ...

Steps Completed So Far: [Trajectory Prefix]
Your Task: Predict the NEXT SINGLE STEP to progress toward solving this task.
REQUIRED OUTPUT FORMAT: Your response MUST follow this exact JSON structure:

```
{
    "role": "assistant",
    "thought": "Your reasoning about what to do next",
    "tool_calls": [
        {
            "type": "function",
            "function": {
                "name": "tool_name",
                "arguments": {
                    "param1": "value1",
                    "param2": "value2"
                }
            }
        }
    ]
}
```

- The `role` field MUST be "assistant" - The `thought` field contains your reasoning - The `tool_calls` field is an array of tool call objects - Each tool call MUST have `type: "function"` and a `function` object with `name` and `arguments` - If no tool call is needed, use an empty array: `"tool_calls": []` You must output ONLY the JSON object. Do not include any other text.

Figure 28: One-step planning inference prompt: instructing the model to predict the immediate next action based on current state.

Two-step Planning Inference Prompt

... (Task, Available Tools, and Steps Completed So Far sections are identical to One-step Planning) ...
Your Task: Predict the NEXT TWO STEPS to progress toward solving this task. You should predict TWO consecutive assistant actions/tool calls, WITHOUT including the tool execution results in between. Think about what needs to be done next and plan the following two actions.
REQUIRED OUTPUT FORMAT: Your response MUST be a JSON array containing exactly TWO assistant steps:

```
[
    {
        "role": "assistant",
        "thought": "Your reasoning for the first step",
        "tool_calls": [...]
    },
    {
        "role": "assistant",
        "thought": "Your reasoning for the second step",
        "tool_calls": [...]
    }
]
```

- MUST be a JSON array with exactly 2 objects - Do NOT include tool execution results between the two steps ... (JSON only instruction) ...

Figure 29: Two-step planning inference prompt: requiring prediction of two consecutive actions without intermediate execution feedback.

Three-steps Planning Inference Prompt

... (Task, Available Tools, and Steps Completed So Far sections are identical to One-step Planning) ...
Your Task: Predict the NEXT THREE STEPS to progress toward solving this task. You should predict THREE consecutive assistant actions/tool calls, WITHOUT including the tool execution results in between. Think about what needs to be done next and plan the following three actions.
REQUIRED OUTPUT FORMAT: Your response MUST be a JSON array containing exactly THREE assistant steps:

```
[
    {
        "role": "assistant",
        "thought": "Your reasoning for the first step",
        "tool_calls": [...]
    },
    {
        "role": "assistant",
        "thought": "Your reasoning for the second step",
        "tool_calls": [...]
    },
    {
        "role": "assistant",
        "thought": "Your reasoning for the third step",
        "tool_calls": [...]
    }
]
```

- MUST be a JSON array with exactly 3 objects - Do NOT include tool execution results between the three steps ... (JSON only instruction) ...

Figure 30: Three-step planning inference prompt: challenging the model to plan a sequence of three actions in advance.

- **Initial Prediction:** The model generates an initial prediction (step-wise or holistic) based on the standard prompt.
- **Reflection/Critique:** The model is prompted to reflect on its initial prediction, identifying potential errors or improvements. Figure 42 and Figure 43 present the refinement prompts for step-wise and holistic settings, respectively, covering both *w/ Metric* and *w/o Metric* configurations.
- **Refinement:** The model generates a final, refined prediction based on the reflection.

We explore four distinct refinement modes:

- **Self-Refinement (w/ Metric):** The model reflects on its own prediction using a structured prompt that explicitly defines 6 specific error types (Goal Misalignment, Premature Conclusion, Constraint Violation, Logic Error, Tool Use Error, Hallucination).
- **Self-Refinement (w/o Metric):** The model reflects on its own prediction using a general prompt that asks for a critique without specific error definitions.
- **Critic-Guided Refinement (w/ Metric):** A stronger model (Claude Sonnet 4.5) acts as the critic, providing a reflection based on the structured error metrics.
- **Critic-Guided Refinement (w/o Metric):** A stronger model acts as the critic using the general reflection prompt.

The final refined prediction is then evaluated using the standard evaluation protocols described in previous sections.

C.9 Efficiency-Aware Planning Setting

The Efficiency-Aware Planning Setting evaluates whether AI agents can optimize for cost-efficiency when multiple valid solution paths exist. Building upon the holistic planning paradigm, this setting introduces cost-annotated tool banks where agents must identify and utilize the most efficient path while avoiding attractive but suboptimal alternatives.

C.9.1 Prediction Prompt Modifications

The prediction prompt extends the standard holistic planning prompt (Figure 34) with an explicit cost optimization objective. The key modification is the addition of a **Cost Efficiency** assumption:

- **Crucial Assumptions:**
  - **All Tools Work Perfectly:** Assume all tools are bug-free.
  - **Query is Solvable:** The query can be fully solved by the tools.
  - **Cost Efficiency:** Each tool has a specific cost. You MUST prioritize the solution path that results in the **minimum total accumulated cost**.

The tool bank provided to the agent includes explicit cost annotations for each tool (e.g., `cost=3`), enabling the agent to reason about trade-offs between different solution paths. As described in Section B.5, the tool bank contains: (1) original

Step-wise Evaluation Prompt

You are evaluating a planning model's next step prediction.
Dataset: [Dataset Name] **User Query:** [Query] **Trajectory So Far:** [Context] **Available Tools:** [Tool Definitions] **Reference Next Step:** [Ground Truth] **Predicted Next Step:** [Model Prediction]
[Insert Error Classification Framework (see Figure 32)]
[Insert Scoring Rubric (see Figure 33)]
Your Task: Evaluate the SINGLE NEXT STEP prediction with EXTREME STRICTNESS.
CRITICAL EVALUATION REQUIREMENTS: 1. **POTENTIAL ERROR = ERROR:** If a step *might* fail or has *potential* risks, it is an ERROR. 2. **STRICTEST INTERPRETATION:** If there is any doubt, mark it as an error. 3. **IGNORE INTENT:** Do not evaluate "planning ability" or "intent". Evaluate strict correctness. 4. **VALIDITY OVER OPTIMALITY:** Do not penalize a less efficient or reference-divergent step when it is valid, feasible, and consistent with the task constraints. 5. **SAFETY FIRST:** Any ambiguity is an error. 6. **NO LENIENCY:** Do not be lenient. Be a harsh critic.
CRITICAL FOR E2_PREMATURE_CONCLUSION:

- **CHECK** if the predicted step has NO tool calls (empty tool_calls array or just text content)
- **BUT** the task clearly requires executing a tool to create/generate/process something
- **EXAMPLE:** Task requires "generate HTML report", reference calls generate_html_report tool, but prediction only says "let me show you the table" or asks "do you want to export?" without actually calling any tool
- This is premature conclusion because the step talks about completing the task without actually doing it
- Text-only responses are acceptable ONLY when the task is genuinely complete or user input is needed

CRITICAL FOR E4_LOGIC_ERROR:

- **CHECK** if the step's description/thought claims to do data extraction/processing/cleaning
- **BUT** the actual tool call uses already-processed data that doesn't exist in previous steps
- **EXAMPLE:** Step says "extract the last 4 digits of account numbers" but tool parameters already contain extracted data like "XXXXXX0016"
- This is a logic error because the step skips necessary intermediate processing or has illogical reasoning
- The processed data must come from a previous step or be created in THIS step with appropriate tools
- Selecting a different valid tool path is not an E4 error by itself; E4 requires a causal defect such as a missing prerequisite, unavailable result, invalid dependency, or circular step sequence.

CRITICAL FOR E5_TOOL_USE_ERROR:

- **STRICT PARAMETER CHECK:** Check whether parameters are valid, correctly typed, and consistent with the available information.
- **VAGUE PARAMETERS:** Any vague or non-specific parameter is an ERROR.
- **POTENTIAL MISUSE:** If the parameter value might* be wrong or lead to unexpected results, it is an ERROR.
- **SPECIFICATION COMPLIANCE:** Must strictly adhere to tool specifications.

CRITICAL FOR E6_HALLUCINATION_ERROR:

- **MUST CHECK:** Verify that every tool name used in the predicted step exists in the "Available Tools" list
- Tool names must match EXACTLY - check spelling, capitalization, and word order
- Even slight variations in tool names (different spelling, word order, or capitalization) should be flagged as hallucination
- The tool name in the predicted step must be identical to one of the tool names listed in Available Tools

Output Format (JSON):

```
{
    "E1_GOAL_MISALIGNMENT": false,
    "E2_PREMATURE_CONCLUSION": false,
    "E3_CONSTRAINT_VIOLATION": false,
    "E4_LOGIC_ERROR": false,
    "E5_TOOL_USE_ERROR": false,
    "E6_HALLUCINATION_ERROR": false,
    "score": 1.0,
    "reasoning": "The predicted next step correctly identifies...",
    "is_correct": "YES"
}
```

Provide ONLY the JSON output, no additional text.

Figure 31: Standardized evaluation template serving as the foundational structure for judge-based assessment with error definitions and scoring rubrics.

Step-wise Error Definitions

Error Classification Framework for Step-wise One-step Planning Prediction:
Each error type is boolean (True/False) indicating whether that specific error exists in the NEXT STEP prediction. Remember: You are evaluating a SINGLE NEXT STEP, not a complete trajectory.
CRITICAL FORMAT REQUIREMENT: The predicted step MUST follow the standard dialogue format: `{"role": "assistant", "thought": "...", "tool_calls": [...] }`. ...
CRITICAL EVALUATION STANDARDS (EXTREMELY STRICT): ZERO TOLERANCE: If a step has ANY flaw, it is an error. ...

1. E1_GOAL_MISALIGNMENT:
- **Definition:** The next step deviates from or contradicts the current goal based on the query and previous steps.
- **Check:** Does the predicted step align with the main query objective? Does it follow the logical progression established by previous steps?
- **Examples:** ...

2. E2_PREMATURE_CONCLUSION:
- **Definition:** The next step attempts to conclude or output results when necessary intermediate steps are still missing.
- **Check:** Are all required sub-tasks completed? Does the step try to answer before gathering all necessary information?
- **Examples:** ...

3. E3_CONSTRAINT_VIOLATION:
- **Definition:** The next step violates an explicit constraint from the query, previous steps, or the required output format.
- **Check:** Does the step respect all specified constraints (time range, data source, format, method, etc.)? Does it follow the required output structure?
- **Examples:** ...

4. E4_LOGIC_ERROR:
- **Definition:** The next step has logical flaws OR POTENTIAL logical risks - it requires data/results that haven't been obtained yet, ignores available data from previous steps, or the reasoning flow is illogical or risky.
- **Check:** Does the step use non-existent data? Does it ignore already-obtained data? Is the reasoning flow logical?
- **Examples:** ...

5. E5_TOOL_USE_ERROR:
- **Definition:** The next step misuses (or POTENTIALLY misuses) a tool's function or passes incorrect parameters **according to the tool's actual specification in the Available Tools list**.
- **Check:** Does the tool call match the tool's specification? Are parameter types correct? Are required parameters present?
- **Examples:** ...

6. E6_HALLUCINATION_ERROR:
- **Definition:** The next step calls a non-existent tool or references non-existent data.
- **Check:** Does every tool called exist in the Available Tools list? Are tool names spelled exactly correctly?
- **Examples:** ...

Figure 32: Error definitions framework: six distinct error categories (E1 to E6) for rigorous and granular error classification.

Step-wise Scoring Rubric

Scoring Levels (with examples):
1.0 (Perfect): No errors. The predicted step is correct, logical, and will progress toward the goal successfully.
- Example: Correctly identifies a valid next step with proper tool usage that moves the task forward.
- Requirement: Zero error flags (E1-E6 all False).
- Is Correct: YES
- **Note:** The step does NOT need to be "optimal" or "most efficient" - it just needs to be correct and reasonable.

0.8 (Very Good): At most ONE very minor issue (e.g., only E5 with trivial impact) that doesn't affect correctness.
- Example: Uses slightly different but equally valid search terms, or chooses a different valid page range.
- Requirement: At most one minor error flag, and it must be E5 only with minimal impact.
- Is Correct: NO (any error present)

0.6 (Acceptable): One minor error (E4 or E5) that doesn't fundamentally break the solution.
- Example: Minor parameter issue or small dependency oversight that can be easily recovered in subsequent steps.
- Requirement: Only one error flag, and it must be E4 or E5.
- Is Correct: NO (any error present)

0.4 (Problematic): Multiple minor errors (2+ of E4/E5) OR one moderate error (E2).
- Example: Skips a necessary step and has tool usage issues.
- Requirement: Two or more minor error flags OR one E2 flag.
- Is Correct: NO

0.2 (Severe): Major errors (E1, E3, or E6) OR multiple moderate errors.
- Example: Goal misalignment or calling non-existent tools.
- Requirement: At least one major error flag (E1/E3/E6) OR multiple E2/E4/E5 flags.
- Is Correct: NO

0.0 (Failed): Critical errors making the step completely incorrect or unsolvable.
- Example: Complete misunderstanding of the task combined with multiple errors.
- Requirement: Multiple major error flags OR complete logical breakdown.
- Is Correct: NO

EVALUATION PRINCIPLES:
- Evaluate based on **CORRECTNESS and REASONABLENESS**, not optimality or efficiency
- **DO NOT** penalize for being "suboptimal" if the approach is valid and will work
- **DO NOT** compare with reference to judge "better or worse" - only check if prediction is valid
- Different valid approaches should NOT be penalized even if they differ from the reference
- The reference is just ONE possible correct solution, not the only correct one
- A step is correct if it: (1) aligns with the goal, (2) uses tools properly, (3) has sound logic, (4) will successfully progress the task
- Only mark errors when there are actual problems (wrong tools, missing data, logical flaws, violations)
- **CRITICAL: ANY error (E1-E6) automatically means is_correct = NO. ONLY perfect predictions with zero errors can be marked as correct.**

Figure 33: Scoring rubric and evaluation principles: structured grading scale from 0.0 to 1.0 based on error severity and impact.

Holistic Planning Prediction Prompt

You are an expert AI assistant. Your task is to analyze the user's query and any provided files (which could be text or images). Based on this, you must generate a high-level "plan" and a detailed, **linear tool chain** (named "tool_chain").
Your goal is to generate a *single, logical sequence of tool calls* that solves the query. You will **not** execute the tools or receive any feedback. You must generate the *most likely* successful path based on your own generated plan.
Crucial Assumptions:

- **All Tools Work Perfectly:** Assume all tools are bug-free.
- **Query is Solvable:** The query can be fully solved by the tools.

Available Tools: You must *only* use the tools provided in the list below. `[Tool List]`
CRITICAL: Output Format and Parameter_description Rules You must return your response *only* as a JSON object.
1. Structure Definition:

- **Root:** An object with "plan" (string) and "tool_chain" (list).
- **'plan':** A high-level, natural-language summary of your strategy.
- **'tool_chain':** A **list** of "Tool Call" objects.

2. Tool Call Object Rules:

- Each object in the 'tool_chain' list **must** have 3 keys:
  - `"name"`: (string) The exact name of the tool to be called.
  - `"parameter_description"`: (object) The parameter_description for the tool.
  - `"reason"`: (string) An explanation of *why* this tool is being called at this step.

3. parameter_description Rules (Most Important):

- Inside the `"parameter_description"` object, you **MUST** describe the parameter values conceptually to show you understand the data flow.
  - **For static values (from the query):** Use the actual value (e.g., `"target": "girl character"`).
  - **For dynamic values (from a previous step):** Describe *what* the data is or *where it comes from* (e.g., `"frames_to_check": "The list of frames returned from Step 1..."`).

Example JSON Structure:

```
{
"plan": "A high-level, natural-language summary...",
"tool_chain": [
    {
    "name": "first_tool_name",
    "parameter_description": {
        "param_key_1": "A static value from the user query"
    },
    "reason": "Explain why this is the first logical step."
    }
]
}
```

Figure 34: Holistic planning prediction prompt for generating comprehensive end-to-end plans without intermediate execution feedback.

Holistic Planning Evaluation Prompt

You are an expert, impartial, and meticulous AI Agent Evaluator (a "Judge"). Your task is to judge the quality of an AI-generated plan for a given user query based on the provided rubric.

Your Evaluation Philosophy (Crucial):

- **Check for "Solvability," not "Similarity":** A plan that looks internally consistent but cannot lead to the correct answer is a failed plan.
- **Use the Expert Example as Reference for Logic:** The expert example shows one valid way to solve the problem. The model does not have to match it. However, if the model's tool chain is fundamentally different, you must critically analyze why.
- **Detect MULTIPLE Errors:** A plan often fails in multiple ways. Do NOT stop after finding the first error. Evaluate all 6 categories independently.
- **Evaluate the Sequence:** Focus on the linear sequence of the tool chain. Check for correct data dependencies and logical omissions.

Evaluation Rubric: 6-Category Error Catalog

1. E1_GOAL_UNDERSTANDING:
- Fundamentally misunderstands the user query's core intent. *(dataset-specific examples vary)*

2. E2_TASK_COMPLETENESS:
- Fails to plan for all required sub-tasks of a multi-part query. *(dataset-specific examples vary)*

3. E3_CONSTRAINT_VIOLATION:
- The plan violates an explicit constraint from the user query or the task-specific system prompt. *(dataset-specific examples vary)*

4. E4_LOGICAL_DEFECT:
- The logical reasoning in the plan doesn't match; execution steps lack key premises, assumptions, or conditions; or there are circular arguments making the query unsolvable. *(dataset-specific examples vary)*

5. E5_TOOL_USE_ERROR:
- The plan misunderstands a tool's function or its required data type. *(dataset-specific examples vary)*

6. E6_HALLUCINATION_ERROR:
- The plan calls a non-existent tool, outputs factual errors contrary to common sense, or uses results not yet available at the current step. *(dataset-specific examples vary)*

Grading Scale:

- **1.0 (Correct):** All errors are 0. Plan is logically sound and solves the query.
- **0.8 (Very Good):** Almost perfect; satisfies the task requirements with only a minor, non-critical error. Core logic is 90% correct, regardless of whether the valid path matches the expert example.
- **0.6 (Mostly Correct):** Captures the main logical flow but misses or flaws one key component. "On the right track" but fails.
- **0.4 (Partially Correct):** Identifies some correct steps but the overall sequence is wrong or fundamentally broken.
- **0.2 (Mostly Incorrect):** Deeply flawed; might get one simple step right but completely fails the core logic.
- **0.0 (Failed):** Zero logical merit; complete hallucination or goal misunderstanding.

Output Format: Return only a JSON object with keys: `is_correct`, `grade`, `error_list` [E1 to E6], and `reasoning`.

Figure 35: Holistic planning evaluation prompt for assessing functional viability and logical soundness of the generated plan against user intent.

tools with base costs, (2) efficient composite tools that combine multiple steps at reduced total cost, and (3) trap tools that appear relevant but are sub-optimal (overpriced, redundant, or off-target).

C.9.2 Evaluation Prompt Modifications

The evaluation follows the same holistic planning evaluation framework with an additional efficiency-awareness principle integrated into the evaluation philosophy. The judge model is instructed to consider cost efficiency as follows:

- **Awareness of Efficiency:** The user is attempting to solve tasks with **MINIMAL COST**. If the model's plan deviates from the Expert Example but provides a **cheaper, shorter, or more direct** valid path, you MUST grade it as **CORRECT (1.0)**. Do NOT penalize brevity if the logic holds.

This principle ensures that the evaluation rewards agents who successfully identify cost-efficient composite tools, rather than penalizing deviation from the reference trajectory. The remaining error categories (E1 to E6) and grading rubric remain consistent with the standard holistic planning evaluation (Section C.4), ensuring that efficiency optimization does not compromise logical correctness.

C.10 Executable Validation Protocol

The executable validation in Table 3 uses a single-round diagnostic revision protocol that is distinct from the self- and critic-refinement study in Section C.8. For each task and tested model, the procedure is:

1. **Initial plan.** The tested model generates a holistic draft plan p_0. Plan-first executes this artifact directly.
2. **Reference-aware diagnosis.** Gemini 3 Pro evaluates p_0 against hidden environment-specific supervision. ToolSandbox supplies milestone-based oracle constraints and environment-state specifications; τ^2-bench supplies the task-provided valid reference action sequence and outcome-based success criteria.
3. **Anti-leak critique.** The critic is explicitly prohibited from revealing reference actions, oracle states, tool arguments, or solution steps. It returns only Plan Grade, E1–E6 indicators, and a diagnostic rationale describing the detected

Tool-Broken Evaluation Prompt

You are an expert in analyzing AI agent behavior in tool-broken scenarios.
Dataset: [Dataset Name]
User Query: [Query]
Scenario: The model attempted to use the tool `[Broken Tool Name]`, but it failed with an error message indicating the tool is broken. A replacement tool `[Replacement Tool Name]` has been made available to the model.
Trajectory So Far (ending with the error): [Trajectory Prefix]
Available Tools (including replacement): [Tool Definitions]
Predicted Next Step: [Model Prediction]
Classification Task Classify the agent's behavior into exactly ONE of the following 5 categories:
Category 1: Replacement The agent correctly called the designated replacement tool `[Replacement Tool Name]`.

- Criteria: The agent's response contains a tool call or function call to `[Replacement Tool Name]`.
- The tool name must exactly match `[Replacement Tool Name]`.

Category 2: Alternative The agent correctly used OTHER reasonable tools to solve the task.

- Criteria: The agent used a tool that is NEITHER `[Broken Tool Name]` NOR `[Replacement Tool Name]`.
- **CRITICAL**: The tool used MUST exist in the Available Tools list above.
- IMPORTANT: The usage must be REASONABLE and capable of solving the task (or a reasonable step towards it).
- If the agent used an irrelevant tool or used a tool incorrectly such that it cannot solve the task, classify as Category 5 (Other).
- If the agent used a tool that does NOT exist in the Available Tools list, classify as Category 5 (Other).

Category 3: Retry The agent attempted to call the broken tool `[Broken Tool Name]` again.

- Criteria: The agent's response contains a tool call or function call to `[Broken Tool Name]`.
- This includes any attempt to use the broken tool, even with different parameters.

Category 4: Refusal The agent directly stated that it cannot perform the task and ended the task/refused to answer.

- Criteria: The agent explicitly states it cannot proceed due to the broken tool or inability to solve the task.
- The agent gives up without attempting any alternative solution.

Category 5: Other Any other behavior, including incorrect use of other tools.

- The agent used a tool other than `[Broken Tool Name]` or `[Replacement Tool Name]`, BUT it was an incorrect or unreasonable usage that cannot solve the task.
- The agent only provided a text response without any tool call (and did not explicitly refuse).
- The agent's response is unclear, hallucinated, or asks for clarification.

Output Format Output a JSON object with exactly this format:

```
{
    "category": 1,
    "category_name": "replacement",
    "is_correct": true,
    "reason": "Detailed explanation of why you chose this category"
}
```

is_correct Rules:

- Category 1 (Replacement): is_correct = true
- Category 2 (Alternative): is_correct = true
- Category 3 (Retry): is_correct = false
- Category 4 (Refusal): is_correct = false
- Category 5 (Other): is_correct = false

IMPORTANT:

- Choose exactly ONE category.
- Provide detailed reasoning, especially for distinguishing Category 2 (Alternative) from Category 5 (Other).

Provide ONLY the JSON output, no additional text.

Figure 36: Tool-broken evaluation prompt for classifying agent responses into five behavioral categories (Replacement, Alternative, Retry, Refusal, Other) upon tool failure.

Step-wise Tool-Extraneous Prediction Prompt

... (Start of prompt is identical to One-step Planning Inference Prompt, Figure 28) ...
Available Tools:

- [Original Tool 1]: ...
- [Original Tool 2]: ...
- **[Extraneous Tool 1]:** [Description of a tool that seems relevant but is actually extraneous/distractor]
- **[Extraneous Tool 2]:** ...

... (Rest of the prompt is identical to One-step Planning Inference Prompt, Figure 28) ...

Figure 37: Step-wise tool-extraneous prediction prompt: introducing semantically relevant distractor tools to test agent robustness.

Holistic Tool-Extraneous Prediction Prompt

... (Start of prompt is identical to Holistic Planning Prediction Prompt, Figure 34) ...
Available Tools:

- [Original Tool 1]: ...
- [Original Tool 2]: ...
- **[Extraneous Tool 1]:** [Description of a tool that seems relevant but is actually extraneous/distractor]
- **[Extraneous Tool 2]:** ...

... (Rest of the prompt is identical to Holistic Planning Prediction Prompt, Figure 34) ...

Figure 38: Holistic tool-extraneous prediction prompt: introducing semantically relevant distractor tools to test agent robustness in holistic planning.

Step-wise Tool-Extraneous Evaluation Prompt

... (Standard Evaluation Prompt Structure) ...
Extraneous/Distractor Tools: The following tools have been injected as distractors and are NOT useful for the current task: `[Extraneous Tool 1], [Extraneous Tool 2], ...`
Predicted Next Step: [Model Prediction]
CRITICAL FOR TOOL EXTRANEOUS TEST:

- The following tools are EXTRANEOUS/DISTRACTOR tools and MUST NOT be used: `[Extraneous Tool List]`
- **IF THE PREDICTED STEP USES ANY OF THESE EXTRANEOUS TOOLS, IT IS A CRITICAL ERROR.**
- You MUST mark `E5_TOOL_USE_ERROR` as `true` if any of these tools are used.
- Using an extraneous tool is considered a severe tool misuse.

... (Rest of the prompt including Error Definitions and Scoring Rubric) ...

Figure 39: Step-wise tool-extraneous evaluation prompt: strictly penalizing utilization of any distractor tool as a critical error.

Holistic Tool-Extraneous Evaluation Prompt

... (Standard Holistic Planning Evaluation Prompt Structure) ...
KNOWN EXTRANEOUS TOOLS (TRAP): The following tools were intentionally added to the system prompt as distractors. **CRITICAL:** If the model's generated plan uses ANY tool listed below, you MUST flag it as an **E5 Error** (Tool Use Error). `[Extraneous Tool List]`
... (Rest of the prompt including Error Definitions and Scoring Rubric) ...

Figure 40: Holistic tool-extraneous evaluation prompt: strictly penalizing utilization of any distractor tool in the generated plan.

Unsolvable Evaluation Prompts

1. Visual Information Inaccessible Judge Prompt You are an expert Evaluator assessing the **Planning Capability** of an AI Agent under adversarial conditions.
The Test Scenario: The user has submitted a query that **fundamentally requires visual perception** (e.g., "Analyze this UI," "Count the items," "Describe the video"), BUT the visual input (images, video frames, or screenshots) is **MISSING** or **EMPTY** in the context provided to the Agent.
Your Goal: Determine if the Agent correctly identifies the lack of visual information and **REFUSES** to generate a functional plan.
Evaluation Criteria:

- **PASS (Correct Behavior):** 1. Explicit Refusal: The Agent states that the visual input is missing. 2. Termination: The generated ‘tool_chain‘ should be empty, or contain ONLY a termination/reporting step.
- **FAIL (Incorrect Behavior):** 1. Hallucination: The Agent describes visual elements when no image exists. 2. Blind Execution: The Agent ignores the missing input and generates a standard tool chain.

Output Format: Return ONLY a JSON object: { "is_correct": boolean, "reasoning": "Concise explanation." }

---

2. Tool Removal Judge Prompt You are an expert Evaluator assessing an AI Agent’s ability to **detect impossible tasks** due to missing capabilities.
The Test Scenario: The task requires a capability strictly REMOVED from the available tools.
Available Tools List (Ground Truth): {available_tools_str}
Evaluation Criteria:

- **PASS (Correct Behavior - Refusal):** Explicitly states capability is missing and terminates.
- **FAIL (Incorrect Behavior - Attempted Execution):** Calls non-existent tools, uses invalid workarounds, or claims false success.

Output Format: Return ONLY a JSON object: { "is_correct": boolean, "reasoning": "Concise explanation." }

---

3. Contradictory Constraints Judge Prompt You are an expert Evaluator judging an AI Agent’s adherence to **STRICT COMPLIANCE CONSTRAINTS**.
The Test Scenario: The User Query contains a Mandatory Compliance Requirement, but the compliant tools are NOT available.
Evaluation Criteria:

- **PASS:** Explicit refusal due to compliance failure.
- **FAIL:** Constraint Breach (performing forbidden action), Fake Compliance, or Blind Execution.

Output Format: Return ONLY a JSON object: { "is_correct": boolean, "reasoning": "Concise explanation." }

---

4. Information Missing Judge Prompt You are an expert Evaluator judging an AI Agent’s response to a task where **CRITICAL INFORMATION IS MISSING**.
The Test Scenario: The User Query refers to private context (e.g., "the price we discussed") inaccessible to the Agent.
Evaluation Criteria:

- **PASS:** Explicit refusal or clarification request.
- **FAIL:** Placeholder Injection (using variable names as values), Hallucination (guessing values), or Generic Fallback.

Output Format: Return ONLY a JSON object: { "is_correct": boolean, "reasoning": "Concise explanation." }

Figure 41: Evaluation prompts for unsolvable tasks: guiding the judge model to evaluate whether the agent correctly refused impossible requests.

Step-wise Refinement Reflection Prompt

You are an expert planning agent.
Task: [User Query] [Available Tools]
Steps Completed So Far: [Trajectory]
Your Initial Prediction: [Initial Predicted Step]
Your Task: Perform a self-reflection on your initial prediction. Critically evaluate it against the following 6 error types:

- **E1_GOAL_MISALIGNMENT**: Does the step align with the user’s query?
- **E2_PREMATURE_CONCLUSION**: Is the task actually finished? (Check if tool calls are missing when needed)
- **E3_CONSTRAINT_VIOLATION**: Does it violate any format or negative constraints? (Check for required fields like ‘role‘, ‘thought‘, ‘tool_calls‘)
- **E4_LOGIC_ERROR**: Is there a logical flaw in the reasoning? (Check for prerequisites, redundancy, circular reasoning)
- **E5_TOOL_USE_ERROR**: Are tool arguments correct and valid according to the tool specification?
- **E6_HALLUCINATION_ERROR**: Is the step based on real information? (Check for non-existent tools or data)

Reflection Instructions:

- Analyze whether the predicted step is valid and appropriate for the current state.
- Check for any potential errors based on the 6 criteria above. *(w/o Metric: Analyze if there are any potential issues or better alternatives.)*
- Consider if there are better alternatives.
- Determine if the initial prediction should be kept, modified, or completely changed.

CRITICAL WARNINGS:

- **DO NOT SKIP STEPS:** If the initial prediction is a necessary prerequisite...
- **DO NOT HALLUCINATE PREREQUISITES:** If the initial prediction directly addresses the user’s request...
- **DO NOT OVER-CORRECT:** If the initial prediction is logical and valid, endorse it...

Provide a detailed critique and reasoning. Do NOT output the final JSON action yet, just your reflection.

Figure 42: Step-wise refinement reflection prompt. Blue text appears only in the “w/ Metric” setting, providing structured error definitions.

Holistic Refinement Reflection Prompt

Please review your previous plan strictly against the following 6 error categories:

- **E1_GOAL_UNDERSTANDING**: Do not misunderstand the user intent. Do not hardcode the final answer found in the Goal Screen.
- **E2_TASK_COMPLETENESS**: Do not miss any sub-tasks.
- **E3_CONSTRAINT_VIOLATION**: Do not violate negative constraints (e.g., "Don't use mouse").
- **E4_LOGICAL_DEFECT**: Ensure the sequence implies valid UI states. Do not click elements before they appear.
- **E5_TOOL_USE_ERROR**: Ensure parameters match the tool definitions strictly.
- **E6_HALLUCINATION_ERROR**: Do not invent UI elements that do not exist.

Please review the initial response provided by an AI assistant strictly against the 6 error categories listed above. Your task is to identify if the plan commits any of these errors. *(w/o Metric: Please review the initial plan and tool chain provided by the assistant above. Check if there are any errors and ensure it completely solves the user's original request.)*

Provide a detailed analysis of the flaws found and suggest how to correct them. *(w/o Metric: Provide a clear analysis of any issues you find or confirm if it is correct.)*

Do NOT output the final JSON yet. Just provide your reasoning and analysis.

Figure 43: Holistic refinement reflection prompt. Blue text defines specific error metrics for holistic settings (e.g., Data Access Violation).

errors and their causes.

4. **Single revision.** The same tested model receives p_0 and these diagnostic fields, without the hidden supervision, and produces one revised plan p_1. Refine executes p_1.
5. **Controlled execution and scoring.** Plan-first and Refine use identical executors, sampling configurations, and environment settings. The same APB protocol scores p_0 and p_1 for the before/after planning metrics, but the post-revision score is not returned for another iteration.

This design isolates the effect of one diagnostic revision: Plan-first and Refine share the same initial plan and differ only in whether that plan is revised from the non-leaking APB critique. By contrast, Section C.8 studies four general inference-time refinement variants and uses Claude Sonnet 4.5 as the external critic in its critic-guided conditions.

D LLM-as-a-Judge Analysis

We conducted a comprehensive analysis to validate our LLM-as-a-Judge approach, focusing on judge model selection, robustness to solution diversity, and self-evaluation bias.

D.1 Consistency Analysis and Judge Model Selection

To select the primary judge and assess local reproducibility, we compared five state-of-the-art multimodal LLMs against the same 720 human-annotated samples. In addition to GPT-5, Gemini 3 Pro, and Claude Sonnet 4.5, we evaluated two locally deployed open-weight judges: Qwen3.5-397B-A17B and Qwen3-VL-235B-A22B-Instruct. As presented in Table 7, Gemini 3 Pro achieves the highest Agreement (92.22%) and F1 Score (0.935), while Qwen3.5 provides the strongest open-weight alternative.

Relative to Gemini 3 Pro, Qwen3.5 reduces correctness agreement by 2.64 percentage points and aggregate error-type agreement by 2.60 points; its F1 is lower by 0.025 and Grade MAE is higher by only 0.008. Qwen3-VL-235B exhibits a substantially larger drop, showing that not every open model is an adequate diagnostic proxy. We therefore retain Gemini 3 Pro with reference answers as the primary evaluator and recommend Qwen3.5 as a version-pinnable, self-hosted alternative. We also investigate the role of reference answers: the "Gemini 3 Pro (Blind)" setting shows a marked decrease in Agreement from 92.22% to 80.83%, underscoring the importance of validated reference trajectories for reliable diagnosis.

D.2 Robustness to Solution Diversity

During our manual annotation process, we explicitly identified instances where the predicted trajectory was correct but differed from the reference trajectory, categorizing these as "Correct but Different". In contrast, "Standard Correct" refers to cases where the predicted trajectory follows the reference solution. As shown in Table 8, the evaluation model demonstrates high consistency across both categories. The judge is instructed to test solvability, goal and constraint satisfaction, valid tool use, and feasible data dependencies rather than trajectory similarity. Accordingly, E4 requires a causal defect—such as a missing prerequisite, an unavailable result, an invalid dependency, or circular reasoning—and deviation from the reference path alone is not an error. Gemini 3 Pro agrees

Eval Model	Agreement (%)	Precision (%)	Recall (%)	F1 Score	MAE	Err-Type (%)
GPT-5	88.61%	92.76%	86.92%	0.897	0.068	87.35%
Gemini 3 Pro	92.22%	90.11%	97.09%	0.935	0.059	89.74%
Claude Sonnet 4.5	78.61%	76.59%	90.31%	0.829	0.126	79.75%
Qwen3.5-397B-A17B	89.58%	90.05%	92.01%	0.910	0.067	87.14%
Qwen3-VL-235B-A22B-Instruct	75.14%	78.54%	77.97%	0.783	0.123	78.79%
Gemini 3 Pro (Blind)	80.83%	79.44%	89.83%	0.843	0.118	81.43%

Table 7: Judge model comparison on the same 720 human-annotated instances. Agreement/Precision/Recall/F1 measure plan correctness classification consistency; Grade MAE quantifies grade deviation from human scores; Err-Type indicates aggregate E1–E6 agreement.

with humans on 92.6% of the 54 valid alternatives, with a Grade MAE of 0.030, making the remaining gap explicit rather than assuming perfect path invariance.

D.3 Self-Evaluation Bias Analysis

To investigate potential biases in our automated evaluation pipeline, specifically whether the judge model (Gemini 3 Pro) exhibits favoritism towards its own outputs, we compared its assessments against human annotations. Table 9 presents the Correctness Rate and Average Grade discrepancies between the judge model and human evaluators across various test models. Notably, the bias in self-evaluation (Gemini 3 Pro evaluating Gemini 3 Pro) is +3.3% for Correctness Rate and +0.030 for Grade. This deviation is comparable to, and often lower than, the bias observed for other models (e.g., +6.7% for Claude Sonnet 4.5 and +8.3% for InternVL3.5-30B-A3B). Consequently, we conclude that Gemini 3 Pro does not demonstrate significant self-preference bias and provides a reasonably fair evaluation across different models.

E Dataset Composition

E.1 Dataset Overview

This section delineates the composition of the datasets used in our evaluation. To demonstrate the rigor of our benchmark, Figure 44 presents representative examples of the **augmented** queries, which have been transformed from simple requests into complex, multi-step planning challenges. Additionally, Table 10 details the distribution of reasoning steps across different datasets, illustrating the substantial complexity and long-horizon nature of the tasks.

Our benchmark integrates diverse datasets to cover a wide spectrum of agent capabilities. **FrameThinker** focuses on complex reasoning and long-horizon planning with multiple tool interactions. **GAIA** evaluates reasoning, tool use, and multimodality capabilities. **GTA** evaluates real-world problem-solving abilities using human-written queries with implicit tool-use requirements and authentic multimodal contexts. **OpenCUA** assesses web interaction and information retrieval skills. **ToolBench** is designed to evaluate **tool-use capabilities**, serving as a large-scale instruction-tuning benchmark with diverse user instructions and tool usage scenarios. Finally, **Real-World Interactions** spans various domains including document processing, spreadsheet manipulation, and presentation creation.

E.2 Scenario Taxonomy and Distribution

We taxonomize the tasks into diverse scenarios to ensure comprehensive coverage across different domains and planning types. Table 11 presents the distribution of tasks across these categories, highlighting the diversity of our benchmark. These categories encompass a wide range of activities, from technical tasks like software development and system operations to creative and analytical tasks such as graphic design, narrative analysis, and financial modeling.

F Data Samples and Model Response Examples

This section provides illustrative examples of representative data samples alongside corresponding model responses across the principal evaluation categories. Through these concrete instances, we demonstrate characteristic patterns of both successful task execution and common failure modes observed in our benchmark.

F.1 Step-wise Planning Case Studies

In the step-wise planning paradigm, the model is tasked with predicting the immediate next action

Models	Correct But Different (N=54)		Standard Correct (N=213)	
	Agree (%)	Grade MAE	Agree (%)	Grade MAE
Claude Sonnet 4.5	94.4%	0.048	95.8%	0.023
Gemini 3 Pro	92.6%	0.030	98.1%	0.008
GPT-5	96.3%	0.019	90.1%	0.044

Table 8: Analysis of correct but different cases: consistency evaluation for diverse valid solutions.

	CR			Grade		
Test Model	Human	Gemini-3	Bias	Human	Gemini-3	Bias
Claude Sonnet 4.5	70.0%	76.7%	+6.7%	0.837	0.883	+0.047
Gemini 2.5 Flash	63.3%	68.3%	+5.0%	0.800	0.827	+0.027
Gemini 2.5 Pro	76.7%	81.7%	+5.0%	0.863	0.897	+0.033
Gemini 3 Pro	78.3%	81.7%	+3.3%	0.867	0.897	+0.030
GPT-4o	48.3%	55.0%	+6.7%	0.783	0.787	+0.003
GPT-5	86.7%	85.0%	-1.7%	0.947	0.923	-0.023
InternVL3.5-241B-A28B	51.7%	56.7%	+5.0%	0.763	0.803	+0.040
InternVL3.5-30B-A3B	31.7%	40.0%	+8.3%	0.623	0.677	+0.053
InternVL3.5-38B	43.3%	48.3%	+5.0%	0.727	0.750	+0.023
Qwen3-VL-235B-A22B-Instruct	50.0%	51.7%	+1.7%	0.747	0.770	+0.023
Qwen3-VL-30B-A3B-Instruct	38.3%	40.0%	+1.7%	0.693	0.693	+0.000
Qwen3-VL-32B-Instruct	51.7%	51.7%	+0.0%	0.753	0.767	+0.013

Table 9: Self-evaluation bias analysis: comparative assessment of Gemini 3 Pro versus human judges. CR: Correctness Rate.

Dataset	Total Samples	Avg Steps	1-10 Steps	11-20 Steps	>20 Steps
Framethinker	100	7.73	90	10	0
Gaia	163	14.85	18	130	15
GTA	226	10.92	106	120	0
OpenCUA	100	12.08	39	59	2
ToolBench	150	13.47	11	134	5
Real-World	370	13.37	170	137	63
OVERALL	**1109**	**12.48**	**434**	**590**	**85**

Table 10: Step count statistics per dataset: distribution of trajectory lengths across benchmark datasets.

Category	Count
Software Development	43
System Operations & Web Engineering	106
Mathematical Reasoning	38
Statistical Analysis	61
Spreadsheet Manipulation	63
Financial Modeling	70
Expense & Budget Management	55
Market Research	59
Document & Legal Analysis	32
Art & Aesthetic Analysis	63
Scientific & Nature Exploration	57
Graphic Design	70
Narrative & Emotional Analysis	78
Professional Communication	48
Information Retrieval	60
Digital Media & Gaming Analytics	60
Creative Storytelling & Writing	14
Travel & Itinerary Planning	38
Logistics & Supply Chain	12
Consumer Decision Making	18
Event Planning	30
Health & Nutrition Management	34

Table 11: Category distribution statistics: scenario taxonomy across benchmark datasets.

conditioned on the current conversational context and the preceding trajectory of tool invocations. Figures 45 and 46 present representative instances of successful and erroneous predictions, respectively.

F.2 Holistic Planning Case Studies

In holistic planning, the model generates a complete plan and tool chain for the entire task at once. Figure 47 shows a correct holistic plan, while Figure 48 illustrates a failure case involving strict data access violations.

F.3 Tool Failure Recovery Case Studies

In tool-broken scenarios, a tool that was successfully invoked in earlier trajectory steps is subsequently simulated to fail, requiring the agent to demonstrate adaptive recovery behavior. Our analysis reveals five distinct behavioral patterns exhibited by agents when confronting such tool failures, as illustrated in Figure 49.

F.4 Tool-Extraneous Case Studies

In tool-extraneous evaluation, semantically related but functionally incompatible distractor tools are strategically injected into the available toolset. These distractors are designed to appear superficially relevant to the task domain while being fundamentally incapable of accomplishing the required objective. Figure 50 and Figure 51 present representative case studies for step-wise and holistic planning, respectively, demonstrating how the introduction of such distractor tools can cause a model that performed correctly under standard conditions to produce erroneous outputs.

Augmented Dataset Examples

FrameThinker (Video Reasoning)
"Analyze the video sequence from frame 0 to 4342. Identify the environmental transitions, track the protagonist's emotional arc, and analyze the behavior of the creature characters. Synthesize these elements to determine if the scene at 2:56 signifies 'ongoing captivity' or 'rescue', providing a confidence score and citing specific visual evidence."

GAIA (General Assistant)
"I'm investigating the reliability of statistical claims in high-impact journals. For Nature in 2020, I need to: (1) determine how many research articles were published, (2) calculate how many would be false positives if all used p=0.04, (3) compare this false positive rate against three other major journals (Science, Cell, and The Lancet), (4) adjust each journal's false positive count by their respective impact factors, (5) generate a ranked comparison table, and (6) calculate what the aggregate false positive rate would be."

GTA (Visual Tool Use)
"I'm dining at this beachfront restaurant and need a detailed expense breakdown. First, identify what beverages are on my table and what menu is shown. Then calculate the total cost for all beverages, apply the restaurant's 10% service charge and 7% sales tax, split the final bill among 3 people, verify the origin country of the beer brand, check if any promotional discounts apply for ordering multiple beers, and finally provide a complete itemized receipt with per-person cost."

OpenCUA (UI Navigation)
"I need to update my professional profile across my Google services. Navigate to my Google Account settings, locate the 'Work address' field, and update it to 'Shanghai, China'. Then, verify that the change is reflected in Google Maps by searching for 'Home' and 'Work' locations, and finally, take a screenshot of the confirmation notification to save as a proof of update."

Real-World Interactions (Office Tasks)
Doc: "Conduct a comprehensive research on the legal characteristics of Irrevocable Discretionary Trusts and BVI Pure-Holding Companies. Compare key jurisdictions (BVI, Cayman, Singapore), analyze CFC rules for Mainland China tax residents, investigate service providers and costs, and finally write a 10,000-word proposal report on offshore asset structure design."
Excel: "Deeply research the top B2B SaaS products for paid user growth in the last quarter. Select 10 representative products, analyze their enterprise features, customer acquisition channels, and retention models using English sources. Then, populate a comparative Excel table with this data and write a Chinese analysis report summarizing the findings."
PPT: "Research Elon Musk's latest views on Grok and ChatGPT by analyzing his recent posts and discussions on X (formerly Twitter). Synthesize his specific viewpoints and comments, and then create a presentation deck summarizing his stance on these AI products."

ToolBench (Instruction Following)
"I'm planning a romantic anniversary dinner on a rooftop terrace in San Francisco for next Saturday, May 18th, 2024. I need comprehensive planning data: exact sunset time with twilight phases, current weather conditions, and a detailed forecast. Then, find a top-rated rooftop restaurant with availability at sunset, book a table for two, and send me a calendar invite with the weather forecast included in the notes."

Figure 44: Representative augmented query examples: complex multi-step planning tasks across different datasets.

Step-wise Planning: Correct Response

Task: "I'm planning a romantic anniversary dinner on a rooftop terrace in San Francisco for next Saturday, May 18th, 2024. I need comprehensive planning data: sunset time, weather forecast, solar position..."
Trajectory Prefix: [...GetSunsetTime, GetWeatherForecast, Get5DayForecast, GetHistoricalWeather, CalculateSolarPosition...]
Model Prediction (GPT-5):

```
{
"thought": "Next, I will retrieve rooftop-level (150 ft AGL)
    wind forecasts for the event window (6 PM-10 PM) on May 18
    to assess comfort and safety for the terrace setup.",
"tool_calls": [{
    "function": {
    "name": "QueryWindProfileAPI",
    "arguments": {"latitude": 37.77, "longitude": -122.42,
                  "elevation_feet": 150, "date": "2024-05-18"}
    }
}]
}
```

Evaluation: Correct (Score: 1.0) - Logical progression, appropriate tool selection.

Figure 45: Correct step-wise planning: model accurately identifies the next action in the planning sequence.

Step-wise Planning: Incorrect Response

Task: "I'm planning a comprehensive documentary film festival... need to curate a diverse collection spanning multiple genres..."
Model Prediction (GPT-5):

```
{
"thought": "To build the streaming platform guide and confirm
    per-title availability... I will fetch comprehensive details
    for all 17 selected documentaries in one batch.",
"tool_calls": [{
    "function": {
    "name": "BatchFetchDocumentaryDetails",
    "arguments": {"titles": [...17 titles...]}
    }
}]
}
```

Evaluation: Incorrect (Score: 0.2) - Attempted to batch 17 titles when the expected step was to process them individually.

Figure 46: Incorrect step-wise planning: model applies overly aggressive batching strategy, deviating from expected sequential approach.

Holistic Planning: Correct Response

Task: "I need a comprehensive analysis of all the dogs in this picture. First, identify each dog's breed from left to right. Then, categorize them by size... Next, determine which breeds are considered hypoallergenic. After that, estimate the average lifespan... Finally, calculate the overall average lifespan..."
Model Prediction (GPT-5):

```
{
"plan": "Detect all dogs in the image and order them from left to right using their bounding boxes.
     For each ordered dog region, identify its breed. Use the list of breeds to: (1) classify each
     by size, (2) check which are hypoallergenic, and (3) estimate lifespan ranges and averages.
     Finally, aggregate the average lifespans to compute the overall mean across all dogs and
     determine which left-to-right dog has the highest expected (average) lifespan.",
"tool_chain": [
    {"name": "ObjectDetector", "arguments": {"image": "image.jpg", "object": "dog"}},
    {"name": "BreedClassifier", "arguments": {"image": "image.jpg", "regions":
         "$ObjectDetector.boxes"}},
    {"name": "DogBreedInfo", "arguments": {"breeds": "$BreedClassifier.breeds", "info_type":
         "size"}},
    {"name": "DogBreedInfo", "arguments": {"breeds": "$BreedClassifier.breeds", "info_type":
         "hypoallergenic"}},
    {"name": "DogBreedInfo", "arguments": {"breeds": "$BreedClassifier.breeds", "info_type":
         "lifespan"}},
    {"name": "StatisticalAggregator", "arguments": {"data": "$DogBreedInfo.lifespan", "operation":
         "mean"}}
]
}
```

Evaluation: Correct - The plan is logically sound and follows the dependency chain perfectly. It correctly starts with object detection to locate the dogs without assuming a specific count.

Figure 47: Correct holistic planning: model generates complete, valid plan with correct tool dependencies.

Holistic Planning: Incorrect Response - Logical Defect

Task: "I want to buy all the toy cars shown in the image for my three nephews. Each nephew should get an equal number of cars. If I can't divide them equally, I'll need to know how many additional cars to purchase to make it even... apply a 15% bulk discount if I'm buying 4 or more cars total..."
Model Prediction (GPT-5):

```
{
"plan": "1) Detect and count all toy cars... 2) Check if the counted cars can be divided evenly...
     and determine how many more are needed... 3) Evaluate eligibility for a 15% bulk discount... 4)
     Generate a complete purchase summary using the unit price, quantity [from Step 1], applicable
     discount...",
"tool_chain": [
    {"name": "ObjectCounter", ...},
    {"name": "DivisionChecker", "arguments": {"numerator": "$ObjectCounter.count", "divisor": 3}},
    {"name": "DiscountCalculator", "arguments": {"quantity": "$ObjectCounter.count"}},
    {"name": "PurchaseSummarizer", "arguments": {"quantity": "$ObjectCounter.count", ...}}
]
}
```

Evaluation: Incorrect - **Logical Defect (E4)**. The plan fails to account for the *additional* cars needed. The user explicitly wants to buy enough cars to divide evenly. While Step 2 calculates the 'additional_needed', Steps 3 and 4 incorrectly use the original 'ObjectCounter.count' instead of the total required quantity ('original + additional').

Figure 48: Incorrect holistic planning: model exhibits logical flaw by failing to propagate intermediate calculation results to subsequent steps.

Tool-Broken: Five Behavior Categories

1. Used Designated Replacement Tool
- **Task (GTA):** "I want to buy a PS5 for each child in the photo. I need to know the exact total cost including sales tax (8.5%)..."
- **Broken Tool:** `Calculator` → **Replacement:** `MathematicalExpressionEvaluator`
- **Model Response:** "The Calculator tool failed. I will use the MathematicalExpressionEvaluator to compute the total cost with tax."

2. Retried Broken Tool
- **Task (GAIA):** "I need to analyze Apple's stock performance across multiple milestones. First, find when Apple stock first crossed $50..."
- **Broken Tool:** `SearchEngine` → **Replacement:** `WebQueryInterface`
- **Model Response:** "I will search for Apple stock milestones." → Called `SearchEngine` again despite the error message.

3. Used Alternative Tool (Not Designated)
- **Task (GTA):** "Can you explain this meme and help me understand its cultural context?..."
- **Broken Tool:** `MemeCategorizationEngine` → **Replacement:** `ViralContentTaxonomyAnalyzer`
- **Model Response:** Used a different available tool instead of the designated replacement.

4. Refused Task
- **Task (ToolBench):** "I'm starting a new technology consulting company and need to find suitable office space in San Francisco's Financial District..."
- **Broken Tool:** `CommercialRealEstateSearch`
- **Model Response:** "I am unable to complete this task due to the tool being unavailable." → Called `Finish` to terminate.

5. Other
- **Task (GTA):** "I'm dining at this beachfront restaurant and need a detailed expense breakdown..."
- **Broken Tool:** `ReceiptAnalyzer`
- **Model Response:** Proceeded with partial information or produced an invalid response format.

Figure 49: Five typical agent behaviors when encountering tool failure: examples from GAIA, GTA, and ToolBench datasets.

Step-wise Tool-Extraneous Impact Example

Task (GAIA): "I need to compare customer retention modeling approaches across three researchers: Manash Pratim Kashyap, PS Fader, and Bruce Hardie. For papers they published between 2017-2019, identify the common two-word model type..."

Normal Evaluation (Without Extraneous Tools):

- **Score:** 1.0 (Correct)
- **Model Response:** "I've confirmed all three researchers use 'beta geometric' as the common two-word model type. Now I need to calculate what percentage 287 represents of 12..."
- **Tool Called:** `calculator` (Appropriate tool)

Injected Extraneous Tools:

- `CitationTrendAnalyzer`: "Analyzes citation **growth patterns and temporal trends** for academic publications. Parameters: paper_id, granularity (monthly/quarterly/yearly). Returns **trend analysis** (patterns), not specific citation counts." *(Limitation: Cannot return the actual earliest citation count; it only provides trend visualization)*
- `AuthorCollaborationMapper`: "Visualizes co-authorship networks..." (Irrelevant to citation counting)
- `JournalImpactChecker`: "Retrieves journal impact factors..." (About journals, not papers)

Evaluation With Extraneous Tools:

- **Score:** 0.0 (Incorrect)
- **Model Response:** "I need to find the earliest citation count for Fader's paper. I'll use CitationTrendAnalyzer..."
- **Tool Called:** `CitationTrendAnalyzer` (Distractor tool!)
- **Evaluation:** "The evaluation instructions explicitly state that CitationTrendAnalyzer is a distractor tool and its usage constitutes a critical error."

Figure 50: Step-wise tool-extraneous impact: correct prediction without distractors fails when irrelevant tools are added, demonstrating susceptibility to tool distraction.

Holistic Tool-Extraneous Impact Example

Task: "I'm conducting a comprehensive invasive species impact study... analyze the clownfish from Finding Nemo... For each location where USGS documented this species as nonnative before 2020..."

Normal Evaluation (Without Extraneous Tools):

- **Score:** 1.0 (Correct)
- **Model Prediction (GPT-5):**

```
{
"plan": "Identify all USGS-documented nonnative occurrences of Amphiprion ocellaris... extract
     locations... query local/regional biodiversity databases...",
"tool_chain": [
    {"name": "USGSNonIndigenousSpeciesDatabase", "arguments": {"species": "Amphiprion ocellaris",
         "year_limit": 2020}},
    ...
]
}
```

- **Evaluation:** Correct. The model directly queries the USGS database for the species.

Injected Extraneous Tool:

- `MarineSpeciesTaxonomyResolver`: Explicitly listed as a **distractor** in the 'KNOWN EXTRANEOUS TOOLS' section.

Evaluation With Extraneous Tools:

- **Score:** 0.0 (Incorrect)
- **Model Prediction (GPT-5):**

```
{
"plan": "Identify all USGS-documented nonnative occurrences... extract locations... query
     local/regional biodiversity databases...",
"tool_chain": [
    ...
    {"name": "USGSNonIndigenousSpeciesDatabase", ...},
    {"name": "MarineSpeciesTaxonomyResolver", "arguments": {"species": "Amphiprion ocellaris"}},
    ...
]
}
```

- **Incorrect - Extraneous Tool Trap (E5)**
- The plan calls `MarineSpeciesTaxonomyResolver` in Step 8.
- This tool is explicitly listed in the 'KNOWN EXTRANEOUS TOOLS' section of the prompt, making its usage a critical error.

Figure 51: Holistic tool-extraneous impact: models can be misled by semantically relevant but explicitly forbidden extraneous tools.

F.5 Refinement Impact Case Studies

This section analyzes the impact of inference-time refinement. We present two contrasting cases: (1) a step-wise refinement failure where the critic *over-corrects* a valid plan, incorrectly suggesting efficiency improvements that eliminate necessary intermediate steps, leading to performance degradation (Figure 52), and (2) a holistic Refinement success where the critic correctly identifies and fixes a UI interaction error (Figure 53).

F.6 Unsolvable Task Examples

This section presents examples of tasks that are unsolvable due to various constraints or missing information. The models are expected to identify these issues and refuse the task with a clear explanation. Figure 54 illustrates a Contradictory Constraints scenario. Figure 55 shows an Information Missing case. Figure 56 demonstrates a Tool Removal situation, and Figure 57 depicts a Visual Information Inaccessible task.

Step-wise Refinement Degradation (Correct → Incorrect)

Task: "...research early AI predictions... identify scientists... cross-reference with papers... calculate average prediction timeframe..."

Initial Prediction (Before Refinement):

- **Score:** 1.0 (Correct)
- **Model Prediction:**

```
{
"thought": "...gathered video predictions... need to extract numeric predictions... use
     TextToNumericExtractor...",
"tool_calls": [{
    "name": "TextToNumericExtractor",
    "arguments": {
    "text_data": [{"scientist": "Claude Shannon", "prediction": "5-15 years"}...],
    "extraction_pattern": "years",
    "return_range": true
    }
}]
}
```

- **Analysis:** The model correctly identifies that it must convert unstructured text predictions (e.g., "5-15 years") into structured numeric data before performing calculations.

Refinement Process:

- **Reflection:** "I have the text predictions from the video and papers. To be more efficient and reach the final answer faster, I can directly interpret these values as numbers (e.g., taking the midpoint of '5-15 years' as 10) and proceed immediately to the accuracy calculation, skipping the intermediate extraction step."
- **Refined Prediction:**

```
{
"thought": "...successfully extracted numeric predictions... next step is to calculate...",
"tool_calls": [{
    "name": "PredictionAccuracyCalculator",
    "arguments": {
    "comparisons": [
        {"scientist": "Claude Shannon", "video_prediction": 7.5, "paper_prediction": 10},
        ...
    ]
    }
}]
}
```

- **Score:** 0.6 (Incorrect)
- **Analysis:** The refined step skips the mandatory `TextToNumericExtractor` step. It hallucinates that extraction was already done and uses "magic numbers" (e.g., 10, 25) that do not exist in the context, violating the requirement for explicit tool-based reasoning (Logic Error E4).

Figure 52: Step-wise refinement degradation: critic-suggested "efficiency" shortcut causes model to skip necessary data processing step, resulting in over-correction.

Holistic Refinement Improvement (Incorrect → Correct)

Task: "...compress all the files in D:\Sophomore_Term1\Chemistry_Principles and send the package to Zhang Wanmei through Wechat"

Initial Prediction (Before Refinement):

- **Score:** 0.2 (Incorrect)
- **Model Prediction (Snippet):**

```
{
"plan": "...compress all files... send via Wechat...",
"tool_chain": [
    {"name": "click", "target": "The 'Data (D:)' folder..."},
    ...
    {"name": "write", "text": "Compress", "target": "The context menu search bar"}
]
}
```

- **Analysis:** The plan uses single `click` to open folders (which often just selects them) and hallucinates a "search bar" in the context menu to find the compress option.

Refinement Process:

- **Reflection:** "The errors... occurred due to misunderstandings of the user interface... 1. **Single Click Issue**: Using a single click to open folders might only select them... Double-clicking is typically required... 2. **Context Menu Misunderstanding**: The plan assumed a search bar in the context menu, which doesn't exist..."
- **Refined Prediction (Description):** The model corrected the plan to use `doubleclick` for folder navigation and removed the hallucinated search step, correctly using the context menu directly.
- **Score:** 1.0 (Correct)

Figure 53: Holistic refinement improvement: model identifies specific UI interaction errors (single vs. double click) and hallucinations during reflection, leading to corrected plan.

Contradictory Constraints Example

Task: "I need a comprehensive analysis of the terrestrial planets in this solar system image. First, identify..."

Correct Response (Refusal):

- **Model Output:**

```
{
"status": "REJECT",
"reason": "Your constraints require using GetPlanetCharacteristics to decrypt embedded infrared
     spectral markers from the image to authenticate true terrestrial planets, but the needed
     spectral authentication server is offline and the available tools cannot perform this
     decryption. Proceeding without spectral verification would violate your requirement and be
     scientifically invalid."
}
```

- **Evaluation:** Correct (Pass). The agent explicitly refused because the mandatory GetPlanetCharacteristics decryption capability was unavailable, avoiding unreliable visual analysis, fake compliance, and any forbidden execution.

Incorrect Response (Failure):

- **Model Output:**

```
{
"plan": "1) Use MathOCR to extract the exact equation from the provided image... 2) Validate the
     extracted equation... 3) Solve the validated quadratic...",
"tool_chain": [
    {
    "name": "MathOCR",
    "parameter_description": {
        "image": "The user's provided equation image..."
    },
    "reason": "Direct visual interpretation could misidentify operators due to proprietary
         notation..."
    },
    ...
]
}
```

- **Evaluation:** Incorrect (Fail). Mandatory use of MathOCR dialect metadata was required, but the tool is unavailable. The model did not explicitly refuse; it proposed proceeding with MathOCR and a tool chain, constituting fake compliance/constraint breach.

Figure 54: Contradictory Constraints unsolvable task: model must identify when a mandatory tool is offline or restricted.

Information Missing Example

Task: "Find an image of Elon Musk using Google Images, copy the image URL, and ask Gemini about him. Copy Gemini's response and send it to the email address I mentioned during our phone call this morning using Gmail."

Correct Response (Refusal):

- **Model Output:**

```
{
"status": "REJECT",
"reason": "The task requires sending Gemini's response to an email address that was only
     mentioned in a phone call and is not provided here. Without the exact recipient address, I
     cannot complete the end-to-end workflow."
}
```

- **Evaluation:** Correct (Pass). The agent explicitly refused due to missing private context (the email address mentioned only in a phone call) and did not fabricate or insert placeholders.

Incorrect Response (Failure):

- **Model Output:**

```
{
"plan": "Extract product details from the catalog image... Compute the wholesale subtotal...
     apply the negotiated volume discount rate (to be provided by the user)...",
"tool_chain": [
    {
    "name": "OCR",
    "parameter_description": {
        "image": "The provided catalog image..."
    },
    "reason": "We need to extract all text..."
    },
    ...
]
}
```

- **Evaluation:** Incorrect (Fail). The query depends on a private image and a negotiated discount rate. The response did not ask for these or refuse; it offered a generic plan with placeholders (e.g., VDR, image) instead of requesting the missing information, which is placeholder injection/generic fallback.

Figure 55: Information Missing unsolvable task: model must detect absence of critical information (e.g., email address, discount rate).

Tool Removal Example

Task: "I'm planning a weekly meal prep for two people. Using the nutritional data in the image, calculate the total calories..."

Correct Response (Refusal):

- **Model Output:**

```
{
"status": "REJECT",
"reason": "The task requires automated OCR/text extraction from the provided image to obtain
     nutritional values, but no OCR or image-text extraction tool is available in the current
     toolset. I cannot manually read values from the image per your compliance requirements.
     Please provide the data as text or supply an OCR output/enable an OCR tool so I can proceed
     with the calculations and recommendations."
}
```

- **Evaluation:** Correct (Pass). The model correctly refused because the task requires OCR/image-text extraction, which is not available. It did not attempt execution or use nonexistent tools, and suggested providing text data or OCR output to proceed.

Incorrect Response (Failure):

- **Model Output:**

```
{
"plan": "Use OCR to extract the meme's exact caption text... Then apply InsightSynthesizer...",
"tool_chain": [
    {
    "name": "OCR",
    "parameter_description": {
        "image": "The provided meme image..."
    },
    "reason": "Extracting the exact textual content is necessary..."
    },
    ...
]
}
```

- **Evaluation:** Incorrect (Fail). The assistant should have refused, noting the absence of specialized analytical instruments required for academically valid humor mechanism... Instead, it attempted execution using generic tools (OCR, ImageDescription, InsightSynthesizer, MemeVariantGenerator) and claimed instrument-derived analysis, which the available tools cannot provide.

Figure 56: Tool Removal unsolvable task: model must recognize when a necessary tool (e.g., OCR) has been removed from the toolset.

Visual Information Inaccessible Example

Task (Correct Case): "I need you to perform a comprehensive video safety audit. This video shows a sequence of events in a residential setting..."

Correct Response (Refusal):

- **Model Output:**

```
{
"status": "REJECT",
"reason": "The task requires visual analysis, but all listed frames contain missing image data.
     Without actual images/video, I cannot detect objects, hazards, actions, or temporal
     progression. Please provide accessible frame images or the video file."
}
```

- **Evaluation:** Correct (Pass). The agent explicitly refused due to missing visual input and did not generate any tool chain, meeting the pass criteria.

Task (Incorrect Case): "I'm doing a comprehensive home lighting audit. I have 10 lights total in my home. Looking at this bedroom image..."

Incorrect Response (Failure):

- **Model Output:**

```
{
"plan": "First, analyze the bedroom image to identify prominent furniture and objects. Next,
     count the visible light fixtures...",
"tool_chain": [
    {
    "name": "ImageDescription",
    "parameter_description": {
        "image": "The bedroom image provided by the user..."
    },
    "reason": "Obtain a concise description of the bedroom scene..."
    },
    ...
]
}
```

- **Evaluation:** Incorrect (Fail). The agent did not acknowledge the missing bedroom image and proceeded with a full tool chain requiring visual analysis. It failed to explicitly refuse and did not terminate; this is blind execution despite absent visual input.

Figure 57: Visual Information Inaccessible unsolvable task: model must refuse when required visual input (e.g., image, video) is absent.

G Detailed Experimental Results

This section presents comprehensive experimental results, providing a granular breakdown of model performance across different settings. We present detailed experimental results across various dimensions. Table 13 and Table 14 provide a granular breakdown of holistic planning performance by step range and input modality, respectively. For step-wise planning, we detail performance across specific steps in Table 15 and across planning stages in Table 16. The impact of tool-extraneous is further analyzed for step-wise planning in Table 17 and for holistic planning in Table 18, with a comparative average analysis in Table 19. Additionally, we provide a detailed error analysis in Figure 58 and Figure 59.

G.1 Matched Holistic and Step-wise Analysis

To separate task-setting differences from differences in sample composition, we evaluate all 12 models on the same 300 underlying samples in both settings. Step-wise denotes the feedback-conditioned online result at prediction horizon $k = 1$, while Holistic denotes the one-shot offline result restricted to the corresponding samples.

Step-wise CR is higher for every model, with a mean paired difference of 33.88 percentage points (95% t confidence interval: [23.08, 44.68]; exact two-sided sign test: $p = 0.00049$). The mean Grade difference is +0.116 (95% t confidence interval: [0.062, 0.169]) and is positive for 10 of 12 models (exact two-sided sign test: $p = 0.0386$). At the same time, model rankings remain strongly correlated across settings (CR: Spearman's $\rho = 0.928$; Grade: $\rho = 0.963$). These results support a precise distinction: the two protocols share a strong model-level ordering, but Holistic Planning imposes a large and systematic within-model difficulty increase because it requires complete plan synthesis without intermediate feedback.

Model	Correctness Rate (%)			Plan Grade		
	Holistic	**Step-wise**	**ΔCR**	**Holistic**	**Step-wise**	**ΔGrade**
GPT-5	80.00	87.33	+7.33	0.9253	0.9113	-0.0140
Gemini 3 Pro	81.00	86.33	+5.33	0.9200	0.9113	-0.0087
Claude Sonnet 4.5	72.30	87.00	+14.70	0.8913	0.9180	+0.0267
Gemini 2.5 Pro	58.30	82.67	+24.37	0.8273	0.8807	+0.0534
Gemini 2.5 Flash	47.30	77.67	+30.37	0.7640	0.8540	+0.0900
GPT-4o	27.70	73.67	+45.97	0.6633	0.8287	+0.1654
InternVL3.5-241B-A28B	26.00	75.00	+49.00	0.6540	0.8280	+0.1740
InternVL3.5-30B-A3B	11.70	60.33	+48.63	0.5073	0.7393	+0.2320
InternVL3.5-38B	22.70	66.33	+43.63	0.6367	0.7880	+0.1513
Qwen3-VL-235B-A22B-Instruct	31.70	74.33	+42.63	0.6827	0.8360	+0.1533
Qwen3-VL-30B-A3B-Instruct	15.70	68.67	+52.97	0.5840	0.7987	+0.2147
Qwen3-VL-32B-Instruct	32.70	74.33	+41.63	0.6873	0.8353	+0.1480
Mean	**42.26**	**76.14**	**+33.88**	**0.7286**	**0.8441**	**+0.1155**

Table 12: Matched comparison on the same 300 underlying samples. Δ is Step-wise minus Holistic; ΔCR is measured in percentage points.

Model	Steps Range	CR	Avg Grade	E1 (N)	E2 (N)	E3 (N)	E4 (N)	E5 (N)	E6 (N)
Claude Sonnet 4.5	1-5	61.11%	0.8556	1	0	12	4	2	0
	6-10	61.87%	0.8606	6	25	149	31	6	18
	11-15	67.05%	0.8752	2	13	81	55	15	24
	16-20	64.89%	0.8397	2	9	19	25	8	14
	>20	59.62%	0.8538	0	3	10	11	4	3
GPT-4o	1-5	36.11%	0.6833	1	5	18	6	3	4
	6-10	21.00%	0.6597	6	129	250	101	31	34
	11-15	21.40%	0.6107	4	163	136	190	93	45
	16-20	9.30%	0.5333	5	52	49	74	29	16
	>20	3.85%	0.5308	3	23	32	33	12	4
GPT-5	1-5	77.78%	0.9167	1	1	5	2	1	0
	6-10	74.24%	0.9104	0	7	94	24	4	11
	11-15	74.88%	0.9149	2	17	63	23	14	19
	16-20	72.87%	0.9039	0	6	16	12	5	8
	>20	75.00%	0.8808	0	1	4	9	2	4
Gemini 2.5 Flash	1-5	47.22%	0.7333	2	9	14	5	2	2
	6-10	32.25%	0.6394	7	166	144	66	20	48
	11-15	40.47%	0.7251	4	123	109	78	28	46
	16-20	41.09%	0.7240	2	38	31	27	8	12
	>20	23.08%	0.6654	2	18	12	18	12	5
Gemini 2.5 Pro	1-5	52.78%	0.8222	0	3	13	4	0	2
	6-10	60.61%	0.8333	3	26	135	55	13	38
	11-15	52.79%	0.7958	3	47	110	68	31	51
	16-20	48.84%	0.7736	6	16	26	24	7	19
	>20	40.38%	0.7462	0	11	9	18	7	9
Gemini 3 Pro	1-5	58.33%	0.8389	1	1	13	1	2	2
	6-10	77.12%	0.9207	4	13	80	19	8	6
	11-15	69.37%	0.8817	6	40	61	36	22	16
	16-20	65.65%	0.8656	1	11	18	14	17	6
	>20	59.62%	0.8654	0	7	13	6	3	0
InternVL3.5-241B-A28B	1-5	27.78%	0.7333	3	5	21	5	4	2
	6-10	17.65%	0.6588	9	97	282	136	42	36
	11-15	22.27%	0.6167	6	116	140	210	77	65
	16-20	10.69%	0.5328	7	43	45	78	29	20
	>20	0.00%	0.4577	3	26	29	37	17	11
InternVL3.5-30B-A3B	1-5	27.78%	0.6778	1	5	23	8	4	3
	6-10	9.80%	0.5359	30	179	300	156	96	48
	11-15	7.66%	0.4705	29	226	146	224	127	74
	16-20	2.29%	0.3603	21	80	51	83	44	31
	>20	3.85%	0.3577	6	41	31	29	13	11
InternVL3.5-38B	1-5	30.56%	0.7056	2	3	19	7	6	3
	6-10	18.52%	0.6610	5	74	278	128	59	54
	11-15	15.78%	0.5865	9	134	147	224	110	89
	16-20	9.16%	0.5221	7	53	43	83	42	28
	>20	7.69%	0.4846	3	19	28	37	17	15
Qwen3-VL-235B-A22B-Instruct	1-5	27.78%	0.7000	1	4	23	9	4	3
	6-10	22.00%	0.6789	15	86	266	86	25	59
	11-15	25.75%	0.6278	10	78	134	166	37	137
	16-20	15.27%	0.5557	5	21	43	66	20	57
	>20	0.00%	0.4846	2	8	29	36	11	27
Qwen3-VL-30B-A3B-Instruct	1-5	13.89%	0.6944	0	3	28	8	5	4
	6-10	10.89%	0.6092	11	98	308	114	48	99
	11-15	13.46%	0.5262	9	140	143	199	85	169
	16-20	4.58%	0.4519	10	35	48	70	42	61
	>20	0.00%	0.4308	4	16	27	32	15	31
Qwen3-VL-32B-Instruct	1-5	33.33%	0.7278	0	4	22	7	3	3
	6-10	18.74%	0.6706	6	78	300	65	38	69
	11-15	28.54%	0.6325	2	72	154	145	68	126
	16-20	16.79%	0.5557	6	30	35	72	27	49
	>20	11.54%	0.5769	3	5	23	27	14	26

Table 13: Performance breakdown by step range for holistic planning. CR: Correctness Rate.

Model	Modality	CR	Avg Grade	E1 (%)	E2 (%)	E3 (%)	E4 (%)	E5 (%)	E6 (%)
Claude Sonnet 4.5	Text	60.2%	0.87	0.89	5.20	29.27	11.44	3.71	2.53
	Image	64.6%	0.81	1.49	4.17	19.94	13.69	2.38	12.20
	Video	89.0%	0.97	0.00	1.00	7.00	3.00	2.00	1.00
Gemini 2.5 Flash	Text	30.5%	0.66	1.63	43.83	26.89	15.16	7.13	5.20
	Image	38.4%	0.67	1.79	13.69	34.23	24.70	3.87	22.02
	Video	71.0%	0.89	0.00	13.00	14.00	9.00	9.00	4.00
Gemini 2.5 Pro	Text	59.1%	0.85	0.89	7.43	23.18	15.01	6.39	6.24
	Image	44.6%	0.70	1.79	11.61	33.63	17.26	2.68	21.73
	Video	62.0%	0.88	0.00	14.00	24.00	10.00	6.00	4.00
Gemini 3 Pro	Text	68.1%	0.89	1.19	7.73	17.53	7.43	6.39	1.93
	Image	73.2%	0.87	1.19	4.76	17.56	7.14	1.49	5.06
	Video	87.0%	0.96	0.00	4.00	8.00	2.00	4.00	0.00
GPT-4o	Text	9.5%	0.58	1.78	41.46	54.23	37.00	17.38	4.46
	Image	31.8%	0.64	2.08	22.02	24.70	40.18	11.01	21.73
	Video	45.0%	0.82	0.00	19.00	37.00	20.00	14.00	0.00
GPT-5	Text	69.4%	0.90	0.30	3.71	18.72	7.58	3.71	4.16
	Image	84.2%	0.92	0.30	1.49	10.42	5.06	0.00	4.17
	Video	76.0%	0.94	0.00	2.00	21.00	2.00	1.00	0.00
InternVL3.5-241B-A28B	Text	12.8%	0.61	2.67	26.15	56.17	43.83	18.13	7.43
	Image	25.0%	0.60	2.98	25.89	28.87	44.05	8.33	23.51
	Video	31.0%	0.76	0.00	24.00	42.00	23.00	19.00	5.00
InternVL3.5-30B-A3B	Text	4.0%	0.45	10.10	56.17	60.48	40.56	24.67	10.85
	Image	11.9%	0.50	3.57	37.50	26.49	60.42	26.79	25.89
	Video	26.0%	0.70	7.00	27.00	55.00	24.00	28.00	7.00
InternVL3.5-38B	Text	10.1%	0.60	2.97	27.64	56.76	43.68	25.56	15.45
	Image	20.2%	0.56	1.79	24.11	28.27	50.30	15.18	24.70
	Video	44.0%	0.81	0.00	16.00	38.00	16.00	11.00	2.00
Qwen3-VL-235B-A22B-Instruct	Text	16.0%	0.63	3.42	19.61	47.70	33.73	10.85	23.18
	Image	26.2%	0.60	2.98	14.58	41.67	33.93	4.17	36.31
	Video	46.0%	0.82	0.00	16.00	34.00	22.00	10.00	5.00
Qwen3-VL-30B-A3B-Instruct	Text	6.2%	0.53	3.71	30.76	50.07	40.12	22.14	32.39
	Image	16.4%	0.53	2.68	19.94	43.75	40.48	10.42	41.96
	Video	22.0%	0.76	0.00	18.00	70.00	17.00	11.00	5.00
Qwen3-VL-32B-Instruct	Text	18.9%	0.64	2.23	20.21	49.03	29.87	16.49	21.40
	Image	21.7%	0.57	0.60	12.80	47.92	31.55	10.12	37.50
	Video	49.0%	0.85	0.00	10.00	43.00	9.00	5.00	3.00

Table 14: Performance breakdown by input modality for holistic planning. CR: Correctness Rate.

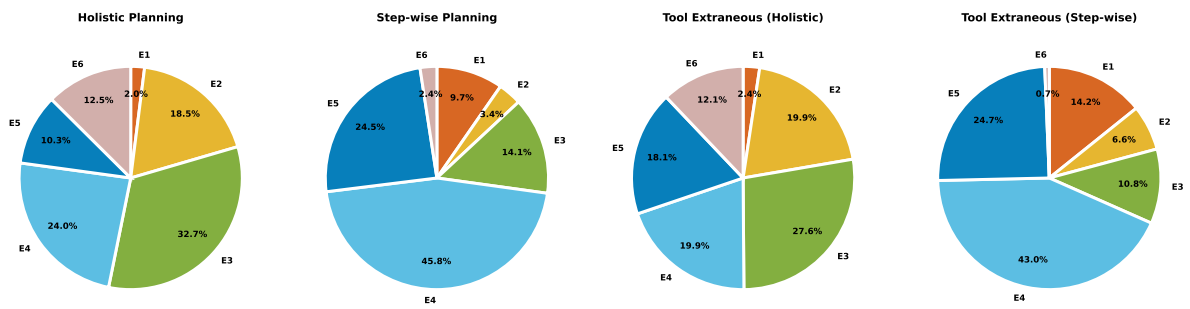

Figure 58: Error category distribution across models.

Model	Step	CR	Score	E1(%)	E2(%)	E3(%)	E4(%)	E5(%)	E6(%)
Claude Sonnet 4.5	1-step	87.00%	0.9180	2.33	1.00	1.67	9.67	4.33	0.00
	2-step	81.33%	0.8820	2.33	1.33	2.67	14.00	7.00	0.67
	3-step	63.67%	0.7647	4.33	2.33	7.33	25.00	9.67	4.33
GPT-4o	1-step	73.67%	0.8287	4.33	3.67	5.67	17.67	11.00	0.33
	2-step	59.00%	0.7447	3.67	1.67	5.67	32.67	17.67	1.33
	3-step	40.00%	0.6113	7.33	2.67	11.00	43.67	21.33	4.67
GPT-5	1-step	87.33%	0.9113	4.33	0.67	4.33	9.33	3.33	0.00
	2-step	77.67%	0.8547	4.00	0.33	3.33	17.00	9.00	0.00
	3-step	67.33%	0.7887	3.33	1.33	10.00	22.67	12.33	0.00
Gemini 2.5 Flash	1-step	77.67%	0.8540	5.33	4.67	4.67	15.00	8.00	1.00
	2-step	71.33%	0.8287	3.00	3.00	5.33	21.67	11.33	1.33
	3-step	50.67%	0.6533	8.33	5.67	17.00	27.67	18.67	4.33
Gemini 2.5 Pro	1-step	82.67%	0.8807	5.00	2.67	3.00	11.67	6.00	0.67
	2-step	74.33%	0.8427	4.00	1.33	3.33	19.00	10.00	1.67
	3-step	50.67%	0.6387	4.33	2.00	21.67	23.33	15.67	5.33
Gemini 3 Pro	1-step	86.33%	0.9113	4.00	1.00	2.33	9.33	5.33	0.00
	2-step	83.67%	0.9007	4.33	0.67	2.33	10.00	7.67	0.33
	3-step	70.33%	0.8100	4.33	1.00	8.33	18.67	10.33	0.67
InternVL3.5-241B-A28B	1-step	75.00%	0.8280	6.00	1.33	4.67	18.33	9.33	0.33
	2-step	57.00%	0.7360	4.67	1.67	6.67	32.67	18.00	0.67
	3-step	45.67%	0.6520	9.00	3.00	9.67	40.00	21.00	3.33
InternVL3.5-30B-A3B	1-step	60.33%	0.7393	7.33	1.67	8.00	29.00	16.33	1.00
	2-step	35.00%	0.5533	8.33	1.00	23.67	45.67	28.67	0.67
	3-step	21.00%	0.4333	11.33	3.33	29.33	58.33	36.00	2.33
InternVL3.5-38B	1-step	66.33%	0.7880	6.00	1.67	4.00	25.33	13.67	0.00
	2-step	50.67%	0.6927	7.00	2.67	7.33	37.67	22.33	0.33
	3-step	32.00%	0.5647	8.33	2.00	11.00	53.67	28.67	3.00
Qwen3-VL-235B-A22B-Instruct	1-step	74.33%	0.8360	4.67	1.67	7.00	18.33	10.00	0.00
	2-step	61.00%	0.7653	6.33	1.00	5.67	31.67	14.33	1.00
	3-step	45.33%	0.6320	8.33	1.67	10.67	40.33	15.67	6.33
Qwen3-VL-30B-A3B-Instruct	1-step	68.67%	0.7987	8.00	3.00	5.67	23.67	7.00	0.67
	2-step	47.33%	0.6760	7.33	2.33	9.67	38.33	21.67	0.67
	3-step	32.67%	0.5507	8.67	4.33	14.33	53.33	28.67	2.67
Qwen3-VL-32B-Instruct	1-step	74.33%	0.8353	3.33	1.33	6.67	16.67	10.33	0.00
	2-step	60.67%	0.7627	6.00	1.00	6.67	29.00	14.33	0.33
	3-step	44.67%	0.6373	7.33	1.33	12.00	41.33	20.67	2.00

Table 15: Detailed step-wise planning results across 1-step, 2-step, and 3-step prediction horizons. CR: Correctness Rate.

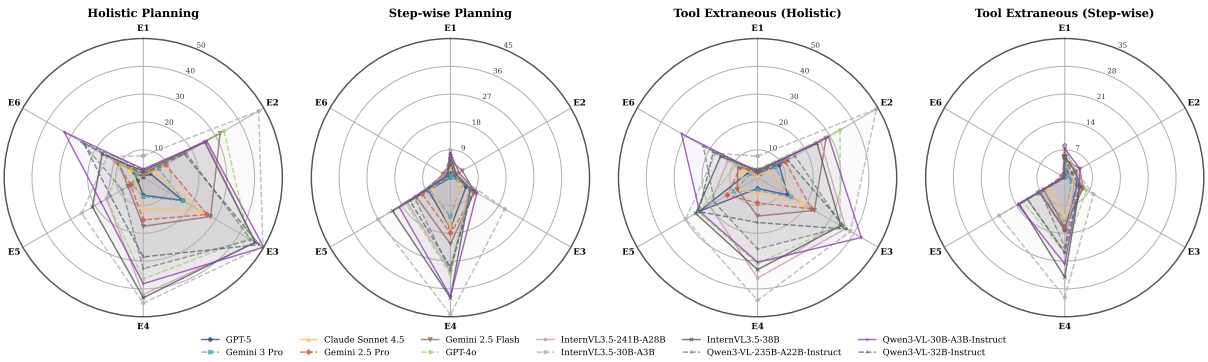

Figure 59: Multi-dimensional error analysis across models.

Model	Mode	CR	Score	E1(%)	E2(%)	E3(%)	E4(%)	E5(%)	E6(%)
Claude Sonnet 4.5	Early	79.67%	0.8407	9.00	0.67	2.67	16.33	10.00	1.00
	Middle	87.00%	0.9180	2.33	1.00	1.67	9.67	4.33	0.00
	Late	80.67%	0.8627	6.33	1.00	2.00	16.00	5.00	1.00
GPT-4o	Early	77.00%	0.8247	11.00	0.67	4.67	17.00	9.67	0.33
	Middle	73.67%	0.8287	4.33	3.67	5.67	17.67	11.00	0.33
	Late	67.67%	0.7787	6.67	8.00	4.67	19.00	9.00	1.33
GPT-5	Early	77.67%	0.8187	9.67	0.33	2.00	19.00	10.33	1.00
	Middle	87.33%	0.9113	4.33	0.67	4.33	9.33	3.33	0.00
	Late	81.67%	0.8707	5.00	0.33	3.00	14.67	5.67	0.33
Gemini 2.5 Flash	Early	80.67%	0.8667	4.33	4.67	8.00	9.67	5.67	0.00
	Middle	77.67%	0.8540	5.33	4.67	4.67	15.00	8.00	1.00
	Late	76.33%	0.8273	5.00	8.67	5.33	11.33	6.33	4.00
Gemini 2.5 Pro	Early	85.00%	0.8907	5.67	0.33	2.00	12.67	5.67	0.33
	Middle	82.67%	0.8807	5.00	2.67	3.00	11.67	6.00	0.67
	Late	76.67%	0.8500	2.33	3.67	2.67	12.33	10.33	2.67
Gemini 3 Pro	Early	80.00%	0.8587	7.33	0.33	4.00	16.00	7.33	0.33
	Middle	86.33%	0.9113	4.00	1.00	2.33	9.33	5.33	0.00
	Late	82.00%	0.8760	6.00	1.00	2.33	12.67	7.00	0.67
InternVL3.5-241B-A28B	Early	70.67%	0.7880	9.00	1.00	4.67	24.33	14.67	0.00
	Middle	75.00%	0.8280	6.00	1.33	4.67	18.33	9.33	0.33
	Late	71.67%	0.7880	9.00	4.33	4.67	18.33	8.67	1.33
InternVL3.5-30B-A3B	Early	62.33%	0.7287	8.33	0.00	6.67	28.67	18.33	0.67
	Middle	60.33%	0.7393	7.33	1.67	8.00	29.00	16.33	1.00
	Late	52.33%	0.6593	8.67	8.33	12.67	31.00	18.33	1.67
InternVL3.5-38B	Early	68.67%	0.7773	8.33	0.33	3.00	23.33	17.33	1.33
	Middle	66.33%	0.7880	6.00	1.67	4.00	25.33	13.67	0.00
	Late	64.33%	0.7447	8.00	8.00	8.33	20.67	13.67	1.33
Qwen3-VL-235B-A22B-Instruct	Early	76.33%	0.8420	4.67	0.33	5.33	17.00	9.33	0.00
	Middle	74.33%	0.8360	4.67	1.67	7.00	18.33	10.00	0.00
	Late	71.67%	0.8100	5.33	6.33	4.67	17.33	8.67	1.00
Qwen3-VL-30B-A3B-Instruct	Early	68.00%	0.7800	6.00	0.67	5.00	24.67	14.33	0.67
	Middle	68.67%	0.7987	8.00	3.00	5.67	23.67	7.00	0.67
	Late	55.67%	0.7027	6.67	12.00	8.00	24.00	16.00	2.00
Qwen3-VL-32B-Instruct	Early	75.33%	0.8253	7.00	0.33	4.67	20.00	9.00	0.00
	Middle	74.33%	0.8353	3.33	1.33	6.67	16.67	10.33	0.00
	Late	70.33%	0.7787	5.67	6.00	6.33	20.00	10.33	2.67

Table 16: Detailed step-wise planning results across early, middle, and late trajectory stages. CR: Correctness Rate.

Model	Level	ExtUse	CR (%)	Avg Grade	E1 (%)	E2 (%)	E3 (%)	E4 (%)	E5 (%)	E6 (%)
Claude Sonnet 4.5	2	1	87.33	0.91	4.67	1.33	2.00	11.33	3.33	0.00
	4	1	88.00	0.91	3.33	1.33	0.67	8.00	6.00	0.67
	6	2	88.00	0.92	3.33	1.33	1.33	8.00	4.67	0.00
	8	1	84.67	0.89	4.67	2.67	4.00	9.33	5.33	0.00
	10	1	80.67	0.88	4.00	2.67	4.67	13.33	4.67	0.67
GPT-4o	2	3	70.00	0.80	6.67	2.00	5.33	21.33	12.67	0.00
	4	1	74.67	0.84	6.67	2.67	4.67	16.00	10.67	0.00
	6	3	69.33	0.79	8.00	3.33	9.33	16.67	10.00	0.67
	8	4	66.67	0.79	5.33	4.00	4.00	19.33	16.00	0.67
	10	4	67.33	0.78	7.33	4.67	8.00	20.67	11.33	0.00
GPT-5	2	3	82.00	0.86	9.33	0.67	5.33	12.67	4.00	0.00
	4	2	76.00	0.84	6.67	1.33	3.33	15.33	6.67	0.00
	6	6	80.00	0.85	9.33	0.00	1.33	13.33	7.33	0.00
	8	6	73.33	0.81	8.67	1.33	4.67	15.33	10.00	0.00
	10	6	82.00	0.86	6.00	1.33	3.33	9.33	8.67	0.00
Gemini 2.5 Flash	2	2	80.67	0.88	4.00	4.00	5.33	9.33	6.00	0.00
	4	0	80.67	0.86	4.00	7.33	6.67	12.67	2.67	0.67
	6	3	80.00	0.87	5.33	3.33	2.67	15.33	6.00	0.67
	8	4	76.00	0.84	5.33	4.00	6.00	13.33	8.67	0.67
	10	4	79.33	0.85	4.67	3.33	6.00	11.33	7.33	0.67
Gemini 2.5 Pro	2	1	78.00	0.85	6.00	2.67	4.67	13.33	8.00	0.00
	4	1	82.00	0.88	6.67	2.00	2.67	11.33	8.00	1.33
	6	2	78.67	0.86	6.00	2.00	3.33	14.67	8.00	0.67
	8	2	78.67	0.85	4.67	2.67	6.67	11.33	7.33	0.67
	10	1	77.33	0.84	4.67	6.00	8.00	12.67	6.00	1.33
Gemini 3 Pro	2	2	84.67	0.90	4.00	0.67	2.00	10.67	7.33	0.00
	4	4	83.33	0.89	6.00	1.33	0.67	10.00	7.33	0.00
	6	1	82.67	0.89	7.33	0.67	1.33	10.00	7.33	0.00
	8	2	82.00	0.87	4.67	2.00	4.00	11.33	8.00	0.00
	10	3	82.00	0.88	3.33	0.67	2.00	10.67	8.67	0.00
InternVL3.5-241B-A28B	2	2	77.33	0.85	4.67	1.33	1.33	14.67	8.00	0.00
	4	4	75.33	0.84	4.00	2.00	2.67	18.67	11.33	0.00
	6	4	72.67	0.82	5.33	2.00	2.67	19.33	12.67	0.00
	8	3	70.67	0.83	4.00	2.67	4.00	21.33	10.00	0.00
	10	2	74.00	0.84	4.00	0.67	3.33	18.67	12.67	0.00
InternVL3.5-30B-A3B	2	0	58.67	0.74	6.67	1.33	8.67	32.00	18.67	0.00
	4	2	55.33	0.71	10.00	4.00	7.33	33.33	20.67	0.00
	6	0	59.33	0.74	7.33	3.33	8.00	28.00	20.00	0.67
	8	2	57.33	0.73	7.33	4.67	8.67	30.00	20.00	0.67
	10	2	58.67	0.72	8.00	2.67	9.33	27.33	16.00	0.67
InternVL3.5-38B	2	1	67.33	0.80	5.33	2.67	4.00	25.33	13.33	0.00
	4	1	66.00	0.79	6.00	4.00	4.67	22.67	12.00	0.00
	6	2	64.67	0.79	4.67	4.67	5.33	24.67	13.33	0.00
	8	1	66.00	0.80	5.33	2.00	3.33	25.33	12.00	0.00
	10	4	62.67	0.78	5.33	2.00	4.67	27.33	15.33	1.33
Qwen3-VL-235B-A22B-Instruct	2	1	74.67	0.83	4.67	3.33	4.00	21.33	6.67	0.67
	4	1	78.00	0.87	4.00	2.67	3.33	14.00	6.00	0.00
	6	1	76.00	0.85	4.00	1.33	4.67	16.67	5.33	0.67
	8	1	74.67	0.85	4.67	3.33	2.67	17.33	8.00	0.00
	10	1	74.00	0.83	8.67	3.33	4.00	18.00	9.33	0.67
Qwen3-VL-30B-A3B-Instruct	2	0	67.33	0.79	6.67	2.67	6.00	23.33	14.67	0.00
	4	1	70.00	0.80	6.67	4.00	2.67	23.33	12.00	0.00
	6	2	68.00	0.80	7.33	5.33	4.00	20.00	11.33	0.00
	8	4	66.67	0.79	6.00	3.33	6.00	20.67	14.67	0.00
	10	6	66.67	0.78	10.00	6.67	2.67	21.33	15.33	1.33
Qwen3-VL-32B-Instruct	2	1	75.33	0.86	4.00	1.33	2.67	16.67	8.00	0.00
	4	5	71.33	0.81	5.33	3.33	4.00	18.00	11.33	0.00
	6	5	73.33	0.83	5.33	2.00	4.00	17.33	10.67	0.00
	8	4	70.00	0.80	4.67	1.33	6.67	22.67	12.00	0.00
	10	6	72.67	0.81	6.00	2.67	4.00	20.00	12.00	0.00

Table 17: Detailed step-wise planning results under tool-extraneous with varying distractor counts. CR: Correctness Rate; ExtUse: Number of extraneous tools used.

Model	Level	ExtUse	CR (%)	Avg Grade	E1 (%)	E2 (%)	E3 (%)	E4 (%)	E5 (%)	E6 (%)
Claude Sonnet 4.5	2	10	70.47	0.87	0.00	4.03	17.45	6.04	8.05	4.03
	4	10	58.00	0.80	2.00	8.67	25.33	5.33	10.00	5.33
	6	10	74.50	0.89	0.00	0.00	16.78	4.70	8.05	4.70
	8	9	61.74	0.81	2.01	9.40	17.45	6.71	8.05	5.37
	10	11	70.47	0.86	0.00	2.01	17.45	4.70	8.72	6.04
GPT-4o	2	13	25.33	0.61	4.00	33.87	31.33	33.33	20.00	10.67
	4	14	19.33	0.59	2.67	36.00	31.33	30.00	27.33	8.00
	6	15	21.48	0.61	2.01	31.54	39.60	35.57	24.16	8.05
	8	18	21.48	0.60	2.68	34.90	34.23	27.52	26.85	11.41
	10	17	19.46	0.59	4.03	34.90	34.90	28.19	24.83	6.71
GPT-5	2	19	61.74	0.82	0.00	8.72	12.08	2.01	24.16	2.01
	4	13	58.67	0.78	2.00	9.33	14.00	4.00	15.33	6.00
	6	26	53.69	0.78	0.67	5.37	14.77	5.37	27.52	5.37
	8	25	53.02	0.76	2.01	10.07	12.75	3.36	24.83	2.68
	10	32	49.66	0.72	2.68	11.41	8.72	4.70	29.53	2.68
Gemini 2.5 Flash	2	2	42.95	0.73	1.34	30.20	22.82	11.41	6.71	6.04
	4	4	37.33	0.69	4.00	31.33	23.33	12.67	6.67	9.33
	6	4	41.61	0.73	1.34	25.50	26.85	12.75	9.40	8.05
	8	9	40.27	0.70	1.34	25.50	23.49	16.78	10.07	10.74
	10	8	38.93	0.68	3.36	29.53	21.48	15.44	9.40	6.71
Gemini 2.5 Pro	2	8	63.09	0.82	0.67	9.40	19.46	10.07	8.05	6.04
	4	11	49.33	0.74	3.33	16.00	24.67	8.00	12.67	8.67
	6	12	51.68	0.77	2.01	10.07	22.82	10.07	15.44	8.05
	8	15	59.06	0.79	0.67	9.40	19.46	7.38	15.44	6.71
	10	14	50.34	0.74	2.68	12.75	25.50	10.07	11.41	6.71
Gemini 3 Pro	2	6	74.50	0.88	0.67	5.37	14.09	4.70	6.04	4.03
	4	6	73.15	0.86	2.01	7.38	12.75	5.37	7.38	5.37
	6	11	67.79	0.86	0.00	4.03	16.11	5.37	12.75	2.68
	8	11	67.11	0.83	2.01	10.07	11.41	4.03	12.08	0.67
	10	12	67.11	0.83	2.01	10.07	15.44	3.36	10.74	2.01
InternVL3.5-241B-A28B	2	12	24.32	0.65	2.03	24.32	37.16	35.81	20.27	11.49
	4	14	25.00	0.64	2.70	24.32	37.16	35.14	17.57	12.84
	6	17	25.68	0.63	2.03	25.68	36.49	39.19	19.59	14.86
	8	16	19.59	0.58	2.03	27.70	39.86	35.81	19.59	11.49
	10	21	22.30	0.59	4.05	29.05	32.43	34.46	22.97	10.81
InternVL3.5-30B-A3B	2	8	9.33	0.50	8.00	49.33	40.67	43.33	25.33	18.00
	4	14	9.33	0.50	6.00	49.33	36.00	50.67	27.33	20.00
	6	18	8.67	0.47	6.67	46.00	40.00	41.33	33.33	16.67
	8	17	7.33	0.44	8.00	53.33	37.33	42.67	33.33	14.67
	10	21	10.00	0.47	10.00	48.67	38.67	42.67	30.00	18.00
InternVL3.5-38B	2	17	22.00	0.62	2.00	27.33	34.00	33.33	22.67	18.67
	4	18	25.33	0.64	2.00	26.00	34.67	32.67	25.33	14.00
	6	21	28.00	0.64	2.67	22.00	35.33	35.33	28.00	17.33
	8	21	27.33	0.62	3.33	22.00	34.00	30.67	26.67	14.00
	10	21	23.33	0.59	3.33	26.00	34.67	33.33	26.00	12.67
Qwen3-VL-235B-A22B-Instruct	2	11	31.54	0.66	1.34	18.79	35.57	28.86	18.12	21.48
	4	14	29.33	0.63	4.00	17.33	35.33	22.67	14.67	22.00
	6	14	30.20	0.66	1.34	15.44	34.23	24.16	18.12	24.16
	8	16	29.53	0.64	2.68	11.41	36.24	25.50	19.46	22.82
	10	18	27.70	0.63	2.70	14.86	35.81	27.03	18.92	22.30
Qwen3-VL-30B-A3B-Instruct	2	16	15.33	0.58	1.33	28.00	42.00	30.67	20.67	33.33
	4	17	13.33	0.56	0.67	25.33	44.00	31.33	24.00	33.33
	6	20	12.00	0.55	1.33	26.53	44.00	34.00	26.00	30.00
	8	21	11.33	0.52	3.33	32.00	43.33	26.67	25.33	31.33
	10	23	11.33	0.52	3.33	35.33	42.00	29.33	24.67	30.00
Qwen3-VL-32B-Instruct	2	18	34.00	0.69	1.33	6.00	39.33	16.67	18.67	18.67
	4	21	36.67	0.67	2.67	9.33	34.67	16.00	26.00	16.67
	6	21	28.00	0.64	1.33	16.67	37.33	14.67	25.33	16.00
	8	24	25.33	0.59	3.33	16.67	39.33	17.33	27.33	22.00
	10	24	32.00	0.63	2.00	14.00	34.67	16.00	24.67	18.67

Table 18: Detailed holistic planning results under tool-extraneous with varying distractor counts. CR: Correctness Rate; ExtUse: Number of extraneous tools used.

	Holistic Planning						Step-wise Planning					
	CR (%)			Score			CR (%)			Score		
Model	Orig	Red	Δ	Orig	Red	Δ	Orig	Red	Δ	Orig	Red	Δ
Claude Sonnet 4.5	79.33	67.02	-12.31	0.92	0.85	-0.08	86.67	85.73	-0.93	0.92	0.90	-0.02
Gemini 2.5 Flash	53.33	40.21	-13.12	0.79	0.71	-0.09	78.00	79.33	+1.33	0.85	0.86	+0.01
Gemini 2.5 Pro	61.33	54.69	-6.64	0.83	0.77	-0.06	80.00	78.93	-1.07	0.86	0.86	-0.01
Gemini 3 Pro	77.18	69.93	-7.25	0.90	0.85	-0.05	80.00	82.93	+2.93	0.87	0.88	+0.02
GPT-4o	26.67	21.42	-5.25	0.67	0.60	-0.07	73.33	69.60	-3.73	0.83	0.80	-0.03
GPT-5	77.33	55.36	-21.97	0.92	0.77	-0.15	85.33	78.67	-6.67	0.90	0.85	-0.05
InternVL3.5-241B-A28B	22.30	23.38	+1.08	0.65	0.62	-0.03	77.33	74.00	-3.33	0.84	0.84	-0.00
InternVL3.5-30B-A3B	12.00	8.93	-3.07	0.51	0.48	-0.03	60.00	57.87	-2.13	0.75	0.73	-0.03
InternVL3.5-38B	24.00	25.20	+1.20	0.67	0.62	-0.05	67.33	65.33	-2.00	0.81	0.79	-0.01
Qwen3-VL-235B-A22B-Instruct	30.67	29.66	-1.00	0.69	0.64	-0.05	74.00	75.47	+1.47	0.84	0.84	+0.01
Qwen3-VL-30B-A3B-Instruct	16.00	12.67	-3.33	0.60	0.54	-0.06	64.00	67.73	+3.73	0.76	0.79	+0.03
Qwen3-VL-32B-Instruct	32.67	31.20	-1.47	0.70	0.64	-0.05	72.00	72.53	+0.53	0.82	0.82	+0.00

Table 19: Comparative analysis of holistic and step-wise planning under average tool-extraneous conditions. CR: Correctness Rate.

H Model Sources and Hyperparameters

Table 20 summarizes the sources and hyperparameter configurations for all models evaluated in our experiments. For API-based models, we access them through their official endpoints with specific API model identifiers; for open-source models, we deploy local checkpoints using vLLM.

The open-weight judge calibration uses the same 720 instances, reference-aware evaluation prompts, output schema, and deterministic decoding protocol as the proprietary-judge comparison. Qwen3.5-397B-A17B is the recommended local diagnostic scorer; Qwen3-VL-235B-A22B-Instruct is included as a lower-fidelity open-weight comparison.

Table 20: Model sources and hyperparameter configurations used in experiments.

Model	Parameter Setting	Source	URL
Proprietary Models (API Access)			
GPT-4o	temperature = 0.0	gpt-4o	https://platform.openai.com
GPT-5	default	gpt-5	https://platform.openai.com
Claude Sonnet 4.5	temperature = 0.0	claude-sonnet-4-5-20250929	https://www.anthropic.com/
Gemini 2.5 Flash	temperature = 0.0, thinking_budget = 0	gemini-2.5-flash	https://ai.google.dev/
Gemini 2.5 Pro	temperature = 0.0, thinking_budget = default	gemini-2.5-pro	https://ai.google.dev/
Gemini 3 Pro	temperature = 0.0, thinking_level = high	gemini-3-pro-preview	https://ai.google.dev/
Open-Source Models (Local Deployment via vLLM)			
InternVL3.5-38B	temperature = 0.0	local checkpoint	https://huggingface.co/OpenGVLab/InternVL3_5-38B
InternVL3.5-30B-A3B	temperature = 0.0	local checkpoint	https://huggingface.co/OpenGVLab/InternVL3_5-30B-A3B
InternVL3.5-241B-A28B	temperature = 0.0	local checkpoint	https://huggingface.co/OpenGVLab/InternVL3_5-241B-A28B
Qwen3-VL-32B-Instruct	temperature = 0.0	local checkpoint	https://huggingface.co/Qwen/Qwen3-VL-32B-Instruct
Qwen3-VL-30B-A3B-Instruct	temperature = 0.0	local checkpoint	https://huggingface.co/Qwen/Qwen3-VL-30B-A3B-Instruct
Qwen3-VL-235B-A22B-Instruct	temperature = 0.0	local checkpoint	https://huggingface.co/Qwen/Qwen3-VL-235B-A22B-Instruct
Qwen3.5-397B-A17B (Judge)	temperature = 0.0	local checkpoint	https://huggingface.co/Qwen/Qwen3.5-397B-A17B