OSWorld, OSWorld2.0

OSWorld: Tianbao Xie et al.
NeurIPS 2024 · Datasets and Benchmarks Track

OSWorld 2.0: Mengqi Yuan, Zilong Zhou, Xinzhuang Xiong et al.
arXiv preprint, 2026

Khalid Khawaji, Ying Tzu Chen, Shreya Dimri

Georgia Tech

A plan must become an action

Example: making a cup of tea. The instructions must become real actions.

PLAN AND TOOLS

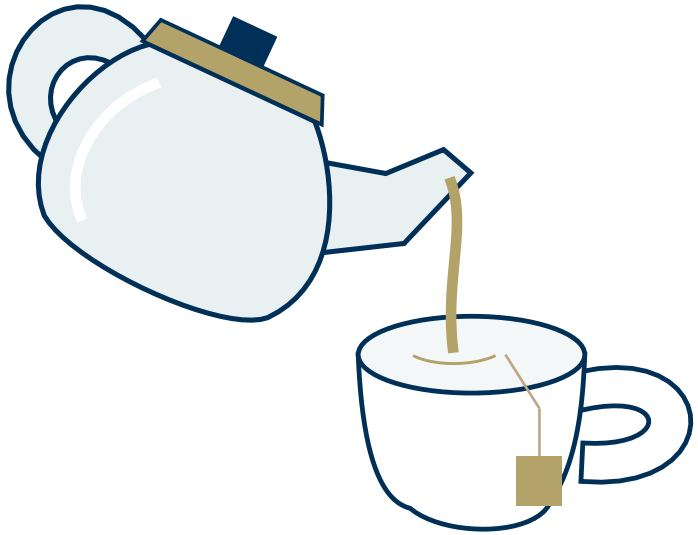
Make a cup of tea.

01 Heat the water.

02 Put tea in the cup.

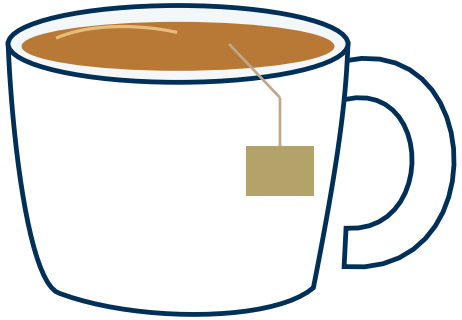
03 Pour and let it brew.

PHYSICAL ACTION



Pour into the actual cup.

FINISHED RESULT



A prepared cup of tea.

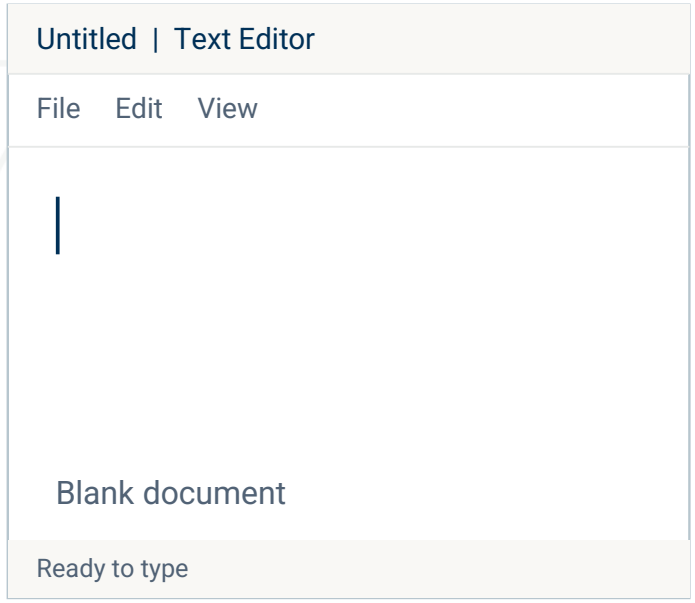
Knowing the procedure is not the same as completing the task.

Georgia Tech

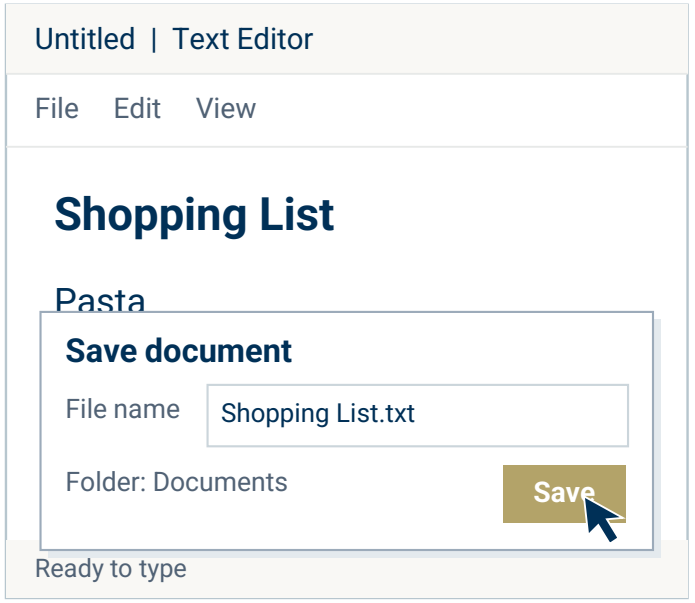
The same problem exists on a computer

Task: create a shopping list and save it in Documents.

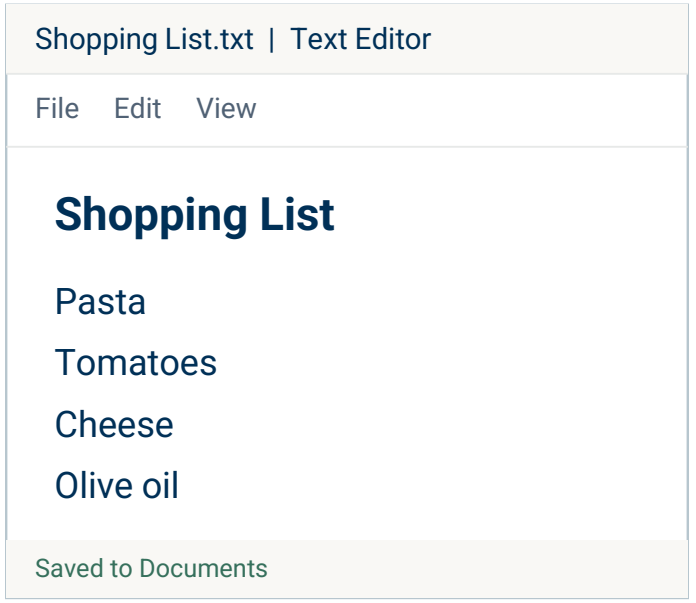
CURRENT SCREEN



MOUSE AND KEYBOARD ACTIONS



SAVED RESULT



Instruction = the goal

Type the list. Click Save.

Outcome = a saved file

The screenshot is the observation. The action happens on the computer.

Georgia Tech

The plan needs the actual situation

A generic answer may not match the current screen or the current progress.

You

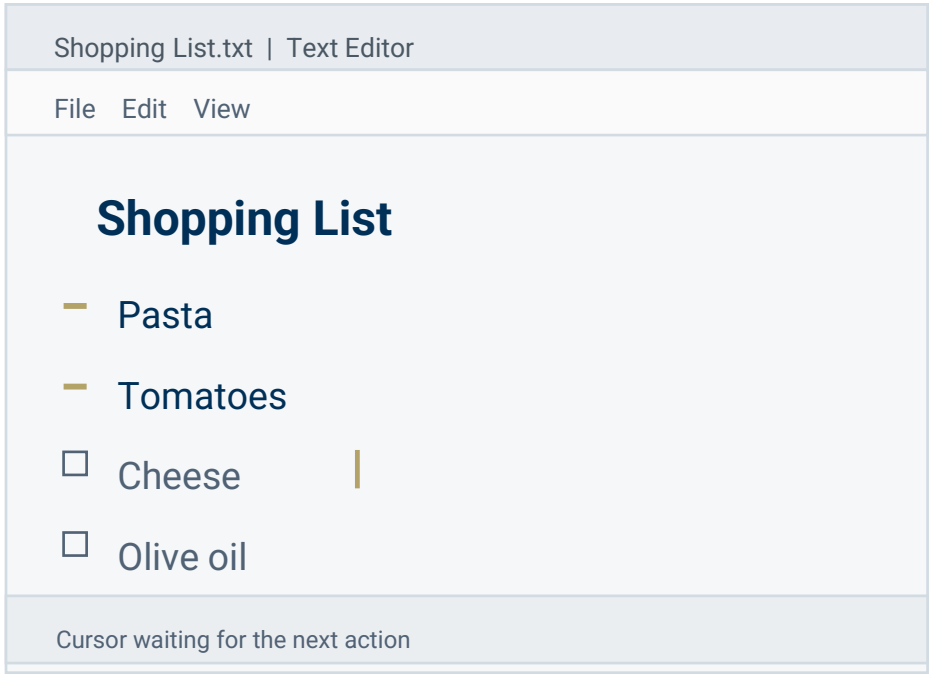
Can you help me save a shopping list in Documents?

ChatGPT

Certainly. A general way is:

1. Open a text editor.
2. Type your shopping items.
3. Choose Save or Save As.
4. Enter a file name.
5. Select the Documents folder.
6. Click Save.

These are useful steps, but the next action depends on what is already on the screen.



The next step depends on the actual situation.

01 **Observe**

What is already typed?
Is the file already named?

02 **Adapt**

Should the agent type another item?
Move the cursor?
Click Save?

Interaction provides the information needed to revise the plan.

Georgia Tech

From Web Benchmarks to OSWorld

What changes: execution scope, task complexity, and evaluation

BENCHMARK COVERAGE

01 **Mind2Web**

Recorded web demonstrations

Predicts web actions • No live execution

02 **WebArena**

Interactive web applications

Executes actions • Selected websites

03 **OSWorld**

Interactive desktop environment

Executes actions • Multiple applications / OS

HUMAN TASK COMPARISON

	OSWorld	WebArena sample
Median time	112s	35s
Human success	72%	88%

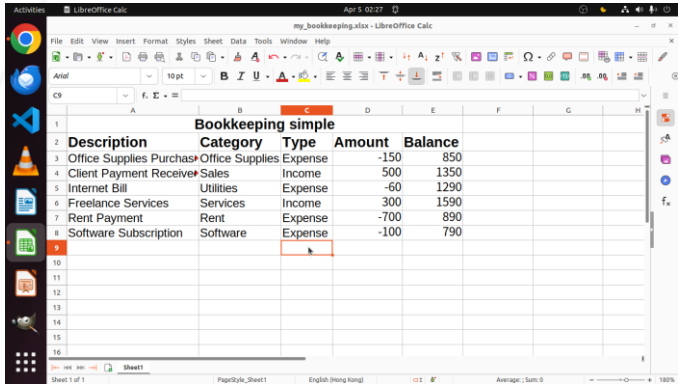
Different task sets: these figures suggest higher difficulty, but are not a direct controlled comparison.

KEY POINT **OSWorld expands evaluation beyond websites to desktop workflows—not every individual task is harder.**

Real work crosses application boundaries

Task: update the bookkeeping sheet using recent receipts.

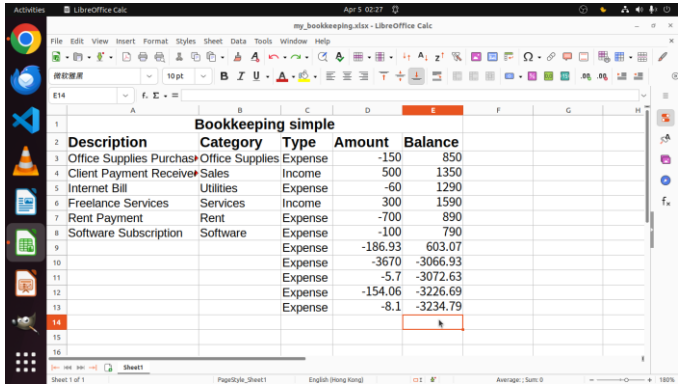
01 OPEN THE SPREADSHEET



02 READ THE RECEIPTS



03 UPDATE THE SHEET



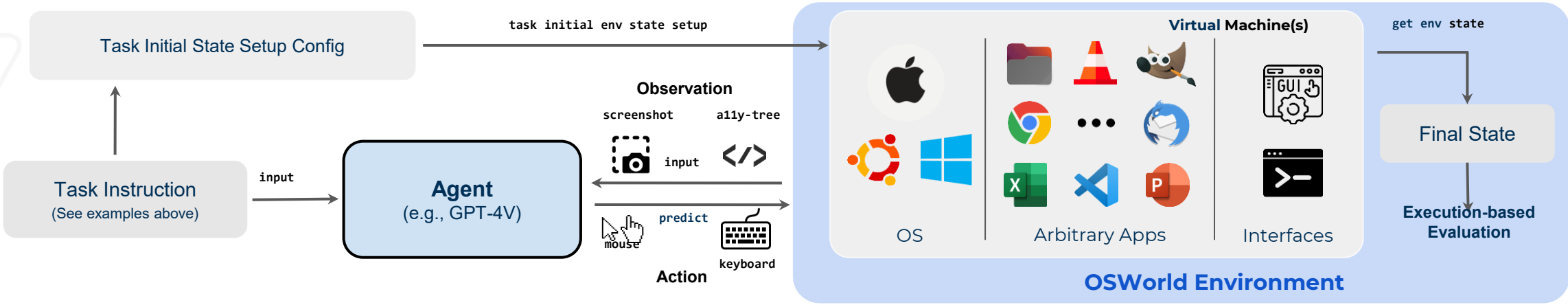
Read the values. Preserve the meaning. Put each value in the correct location.

Success requires the whole workflow, not just one correct click.

Georgia Tech

OSWorld connects the agent to a computer

The original environment diagram: preparation, interaction, and final-state evaluation.



01 Prepare

Set up the task state.

02 Interact

Execute and observe.

03 Evaluate

Check the resulting output.

The model chooses actions inside an agent loop. OSWorld runs the computer and evaluates the result.

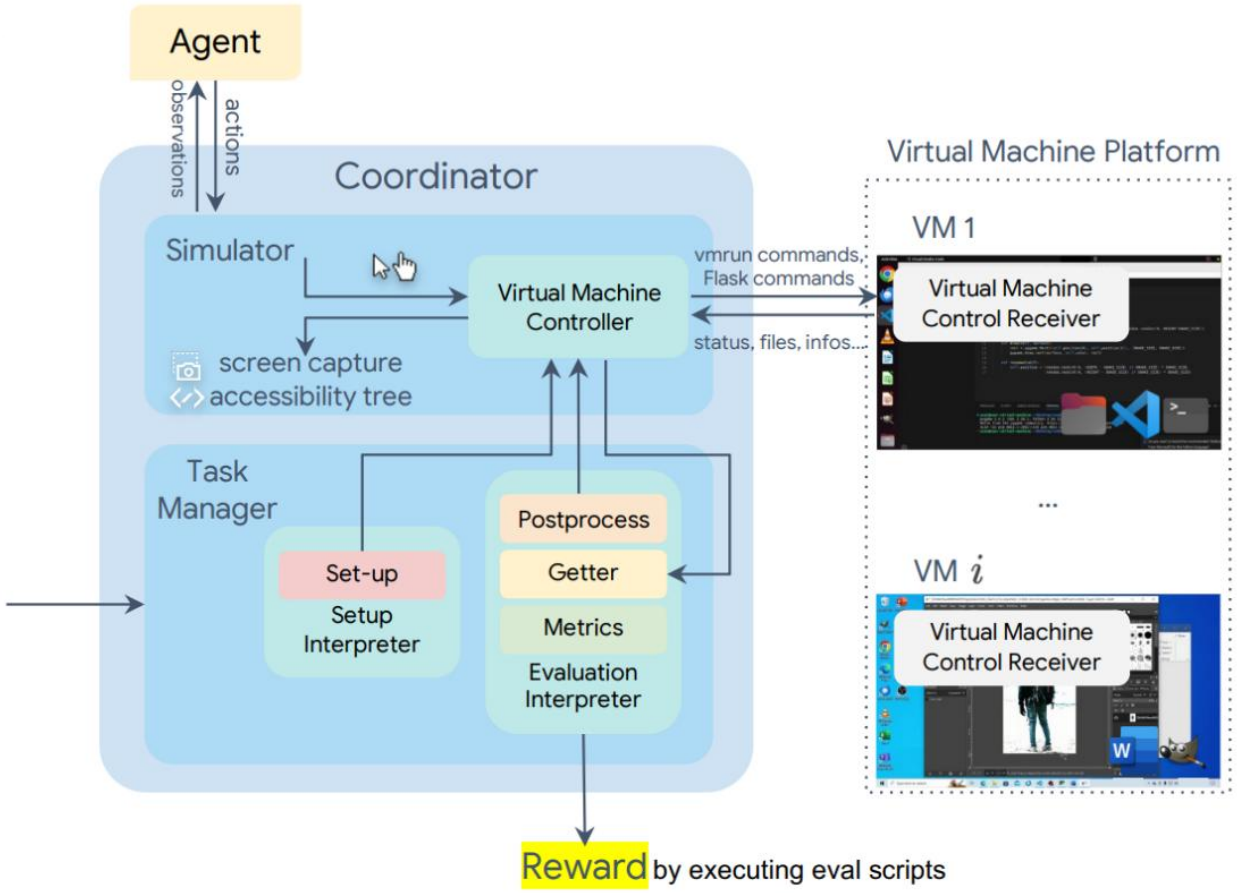
Georgia Tech

OSWorld System Architecture

How the agent interacts with the computer and how OSWorld checks the result.

Config

```
{ "instruction": "Please update my bookkeeping sheet with
the recent transactions from the provided folder, detailing
my expenses over the past few days.",
  "config": [{"type": "download",
      "parameters": {"files": [
{"path": "/home/user/Desktop/my_bookkeeping.xlsx",
 "url": "https://drive.google.com/uc?id=xxxx"},
{"path": "/home/user/Desktop/receipt_0.jpeg",
 "url": "https://drive.google.com/uc?id=xxxx"},…]}},
    {"type": "open",
      "parameters": {  "path":
"/home/user/Desktop/my_bookkeeping.xlsx"}}],
   "evaluator": {"postconfig": [{"type": "activate_window",
      "parameters": {"window_name": "my_bookkeeping.xlsx -
LibreOffice Calc",... ],
    "result": {"type": "vm_file",
      "path": "/home/user/Desktop/my_bookkeeping.xlsx",
      "dest": "my_bookkeeping.xlsx"},
    "expected": {"type": "cloud_file",
      "path": "https://drive.google.com/uc?id=xxx",
      "dest": "my_bookkeeping_gold.xlsx" },
    "func": "compare_table",
    "options": {
      "rules": [{
          "type": "sheet_fuzzy",
          "sheet_idx0": "RNSheet1",
          "sheet_idx1": "ENSheet1",
          "rules": [ {"range": [ "A1:A8",... }]}]
}
```



Four ways to observe the interface

Same computer screen, four types of information for choosing an action.

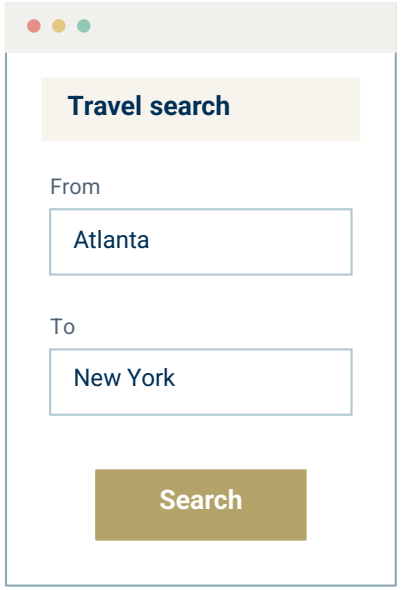
1 Tree

Structured elements only

```
Window: Browser
  Heading: Travel search
  Textbox: From
    Value: Atlanta
  Textbox: To
    Value: New York
  Button: Search
```

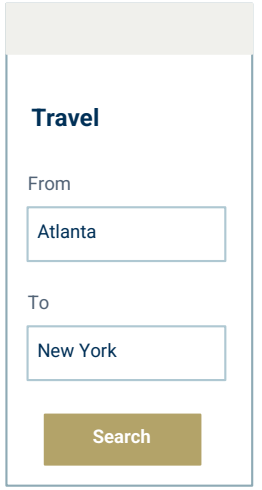
2 Screenshot

Pixels only



3 Screenshot + tree

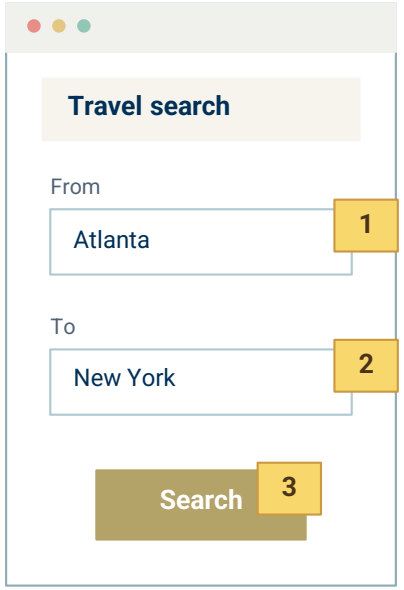
Visual and structured




```
Window: Browser
  Heading: Travel search
  Textbox: From
    Value: Atlanta
  Textbox: To
    Value: New York
  Button: Search
```

4 Set-of-Marks

Numbered UI elements



RESEARCH QUESTION How much vision is needed when the interface tree is available?

Georgia Tech

The task is partially observable

Partially observable Markov decision process: (S, O, A, T, R).

S	**State**	The actual condition of the computer.
O	**Observation**	The information the agent receives.
A	**Action**	A mouse or keyboard operation.
T	**Transition**	How the state changes after an action.
R	**Reward**	The task-dependent outcome score.

The screen observation is not the complete computer state.

Georgia Tech

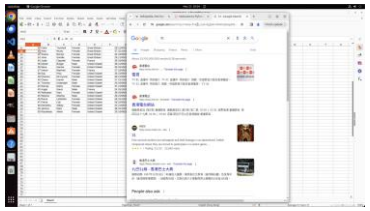
Actions control the real interface

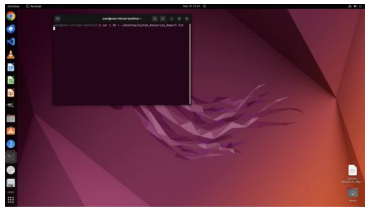
The agent produces code. OSWorld executes it on the computer.

Function	Description
`moveTo(x, y)`	Moves the mouse to the specified coordinates.
`click(x, y)`	Clicks at the specified coordinates.
`write('text')`	Types the specified text at the current cursor location.
`press('enter')`	Presses the Enter key.
`hotkey('ctrl', 'c')`	Performs the Ctrl+C hotkey combination (copy).
`scroll(200)`	Scrolls up by 200 units.
`scroll(-200)`	Scrolls down by 200 units.
`dragTo(x, y)`	Drags the mouse to the specified coordinates.
`keyDown('shift')`	Holds down the Shift key.
`keyUp('shift')`	Releases the Shift key.
`WAIT`	Agent decides it should wait.
`FAIL`	Agent decides the task is infeasible.
`DONE`	Agent decides the task is finished.

AN EXECUTABLE ACTION

```
pyautogui.click(x, y)
```





One interaction step may contain several low level operations.

WAIT, FAIL, and DONE control the task process. They are not ordinary mouse clicks.

Georgia Tech

Observe, act and inspect the result

Original example: monitor CPU activity for 30 seconds and save the output.

OBSERVATION → ACTION → NEW OBSERVATION



01 Open the terminal **02 Focus the window** **03 Run the command** **04 Inspect the output**

A repeated interaction loop does not by itself mean that model weights are updated.

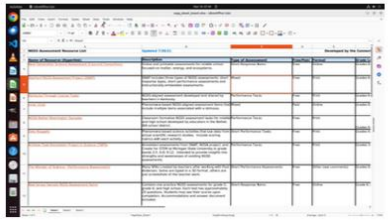
Georgia Tech

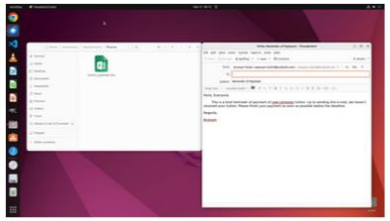
The evaluator checks the outcome

Execution-based checking: inspect the saved state, not a scripted sequence of clicks.

Initial State	Task Instruction	Evaluation Script (Simplified)

	*Can you help me clean up my computer by getting rid of all the tracking things that Amazon might have saved?*	`cookie_data = get_cookie_data(env)`
		`rule = {"type":"domains",`
		`"domains":[".amazon.com"]}`
		`is_cookie_deleted(cookie_data, rule)`

	*Rename "Sheet 1" to "LARS Resources". Then make a copy of it. Place the copy before "Sheet 2" and rename it by appending a suffix "(Backup)", ...*	`result = get_file(env)`
		`expected = get_file(cloud)`
		`rules = [{"type":"sheet_name"},`
		`{"type":"sheet_data",`
		`"sheet_idx0":0,`
		`"sheet_idx1":1}...]`
		`compare_table(result, expected, rules)`

	*I've drafted an e-mail reminder for those who haven't paid tuition. Please help me to check out their e-mails from the payment record and add to the receiver field.*	`tree = get_a11y_tree(env)`
		`rules = [{"selectors":`
		`["tool-bar[attr|id=MsgHeadersToolbar]`
		`label[name=To]`
		`[attr|class=\"address-pill\"]>`
		`label[attr|class=\"pill-label\"]`
		`[name*=\"fox@someuniversity.edu...]`
		`check_a11y_tree(tree, rules)`

01 **Retrieve**

Collect files, cookies, or UI state.

02 **Compare**

Use a task-specific reference or rule.

03 **Score**

Check the outcome (sometimes partial).

KEY DISTINCTION **The evaluator knows the reference; the agent may not.**

Georgia Tech

The benchmark contains 369 computer tasks

Real applications. Intermediate starting states. Custom outcome checks.

369

Ubuntu tasks

101 cross-app workflows

43 additional Windows tasks for analysis

Table 3: Key statistics in OSWORLD. The "Supp. tasks" refers to the Windows-based tasks, that could only be used after activation due to copyright restrictions.

Statistic	Number
Total tasks (Ubuntu)	369 (100%)
- Multi-App Workflow	101 (27.4%)
- Single-App	268 (72.6%)
- Integrated	84 (22.8%)
- Infeasible	30 (8.1%)
Supp. tasks (Windows)	43
Initial States	302
Eval. Scripts	134

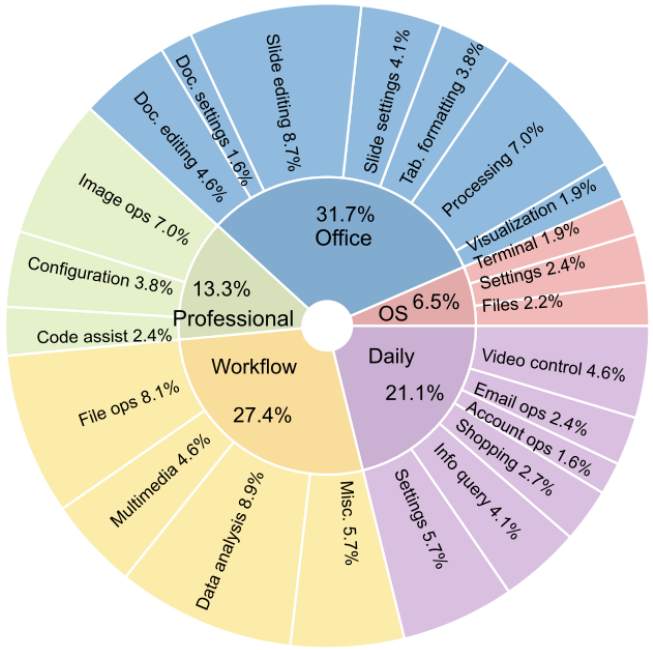


Figure 3: Distribution of task instructions in OSWORLD based on the app domains and operation types to showcase the content intuitively.

The collection includes 30 infeasible tasks and 134 evaluation functions.

Georgia Tech

Before comparing scores, compare the setups

The 2024 baselines vary along three axes; the results are not a pure model size ranking.

01 / MODELS

Different models

Open models include Mixtral-8×7B and Llama-3-70B. The evaluation also uses GPT, Gemini, and Claude models.

02 / PERCEPTION

Different inputs

Some agents receive accessibility trees, some screenshots, and others combined inputs or Set-of-Marks.

03 / ACTIONS

Same step cap

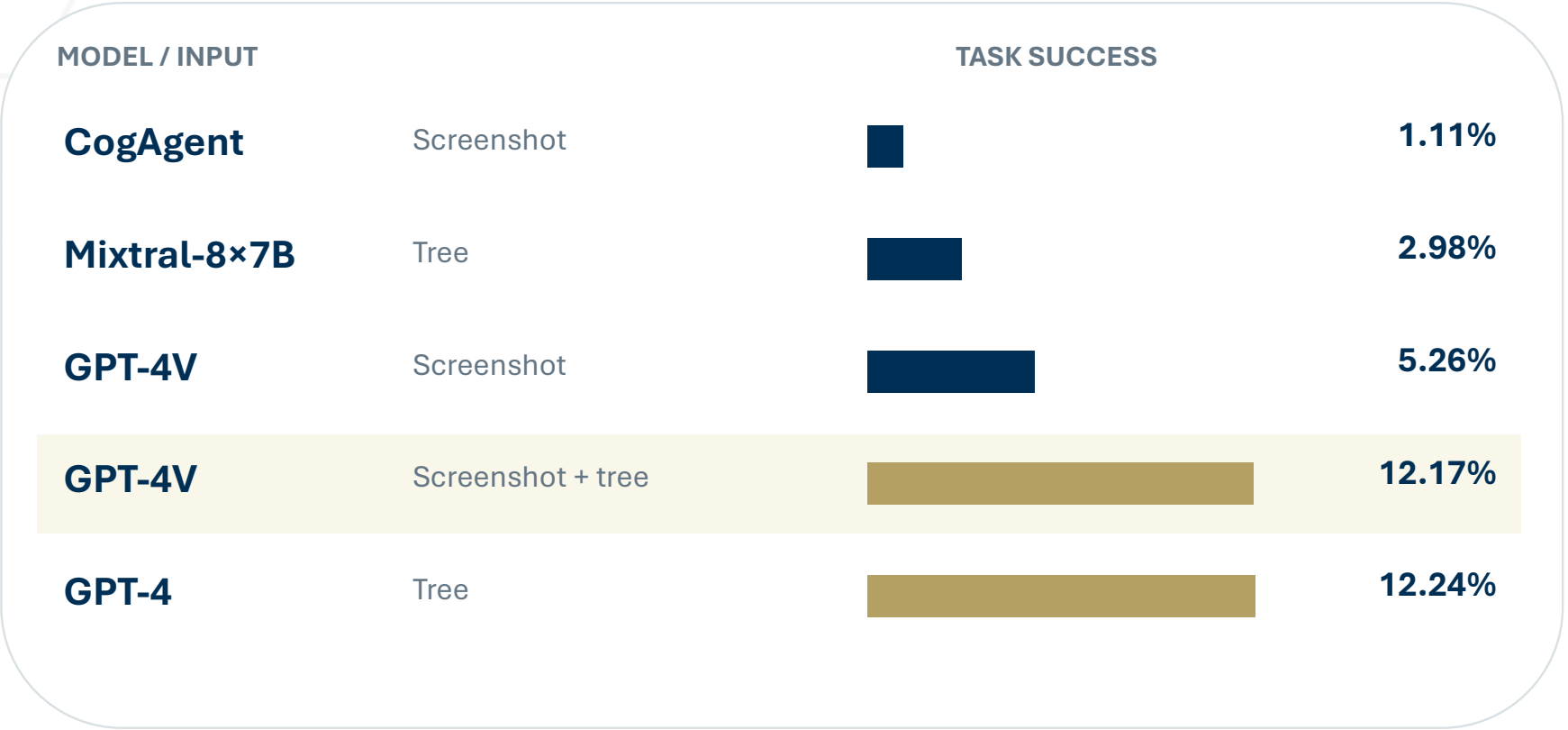
Prompted agents choose computer actions under a maximum of 15 interaction steps. History is limited.

KEY FACTORS **Model capability interface information and action budget all affect task success.**

Georgia Tech

Baseline success: stronger inputs still leave a large gap

Representative historical settings on the main Ubuntu benchmark (not a current leaderboard).



HUMANS

72.36%

BEST AGENT

12.24%

GPT-4 + tree

KEY RESULT **The GPT-4V gain with a tree (+6.91 points) is as important as the model label.**

Georgia Tech

Does structured input always help a visual agent?

Original Table 5: compare each model against itself, with the same Ubuntu benchmark.

		SCREENSHOT	SCREENSHOT + TREE	
GPT-4V	The tree more than doubles its success rate.	**5.26%**	**12.17%**	**+6.91 pts**
Gemini-ProV	Adding a tree makes this model less successful.	**5.80%**	**3.48%**	**−2.32 pts**

VISION QUESTION How much does visual input help agents complete computer tasks?

Georgia Tech

Better vision depends on the input format

Original Figure 5: image-resolution ablation on a 10% task subset.

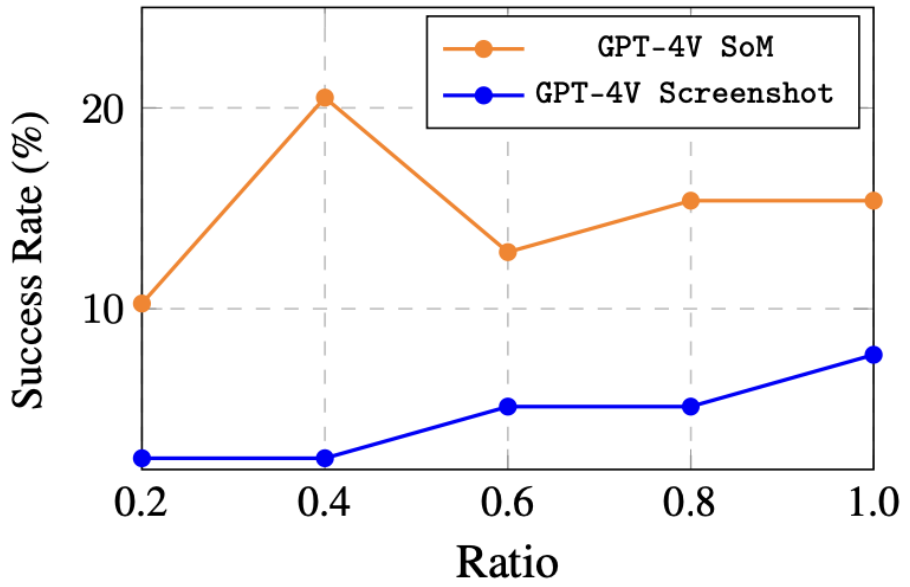


Figure 5: The effect of downsampling on the screenshot on performance with down-sampling ratios of 0.2, 0.4, 0.6 and 0.8 and run on a subset (10%) of examples.

01 **Pixels matter**

Screenshot-only performance generally improves with resolution.

02 **Not a uniform trend**

The Set-of-Marks curve peaks at an intermediate ratio.

RESOLUTION More pixels help screenshot-only agents here, but not every input method.

Georgia Tech

Useful history is more than extra screenshots

Original Figures 6–7: observation length and trajectory-history analysis.

HOW LARGE IS ONE OBSERVATION?

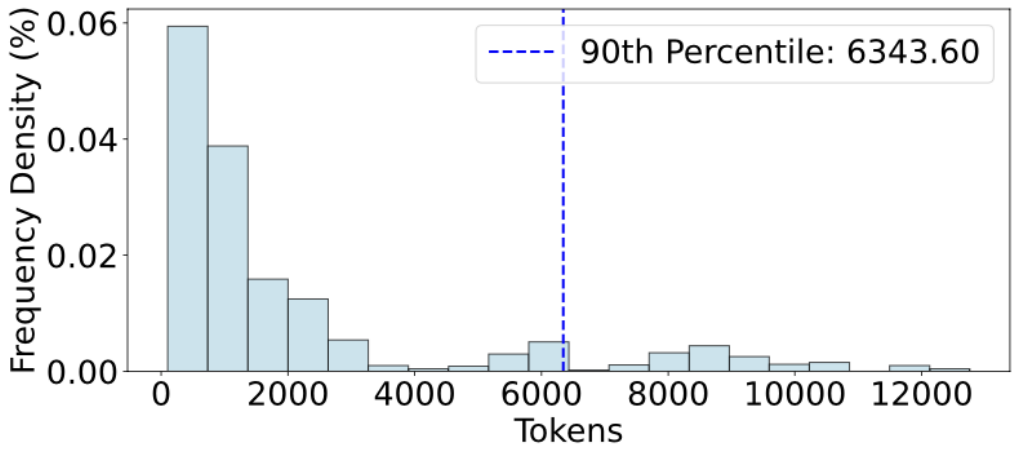


Figure 6: The length distribution of a11y tree as observation from sampled trajectories.

DOES MORE HISTORY HELP?

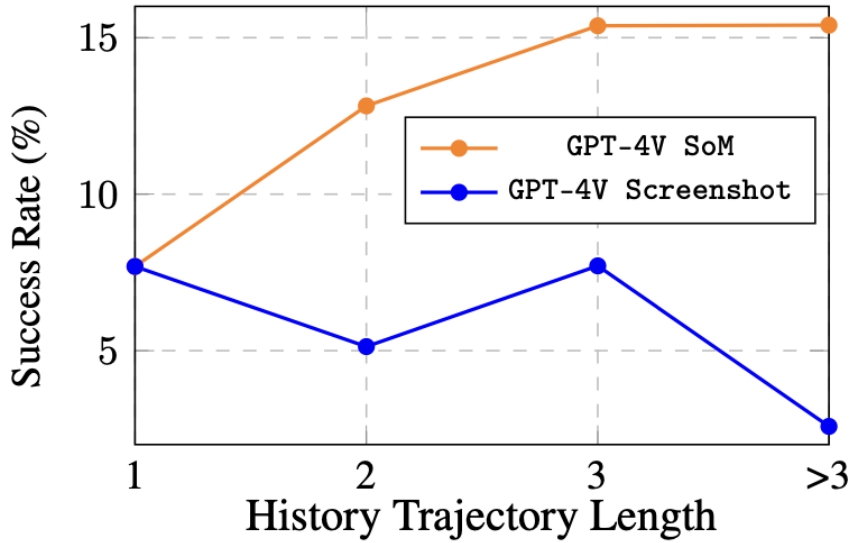


Figure 7: The effect of length of history on performance with the history encoding length of 1, 2, 3, and > 3 and run on a subset (10%) of examples.

HISTORY EFFECT More history helps SoM, but extra screenshots alone can hurt.

Georgia Tech

A similar pattern appears across operating systems

Original Table 7. Screenshot-only GPT-4V on adapted Ubuntu and Windows tasks.

Table 7: Comparison of model performance and correlation across operating systems.

OS	SR (%)	Correlation Coefficient
Ubuntu	4.88	0.7
Windows	2.55	

REPORTED CORRELATION

0.7

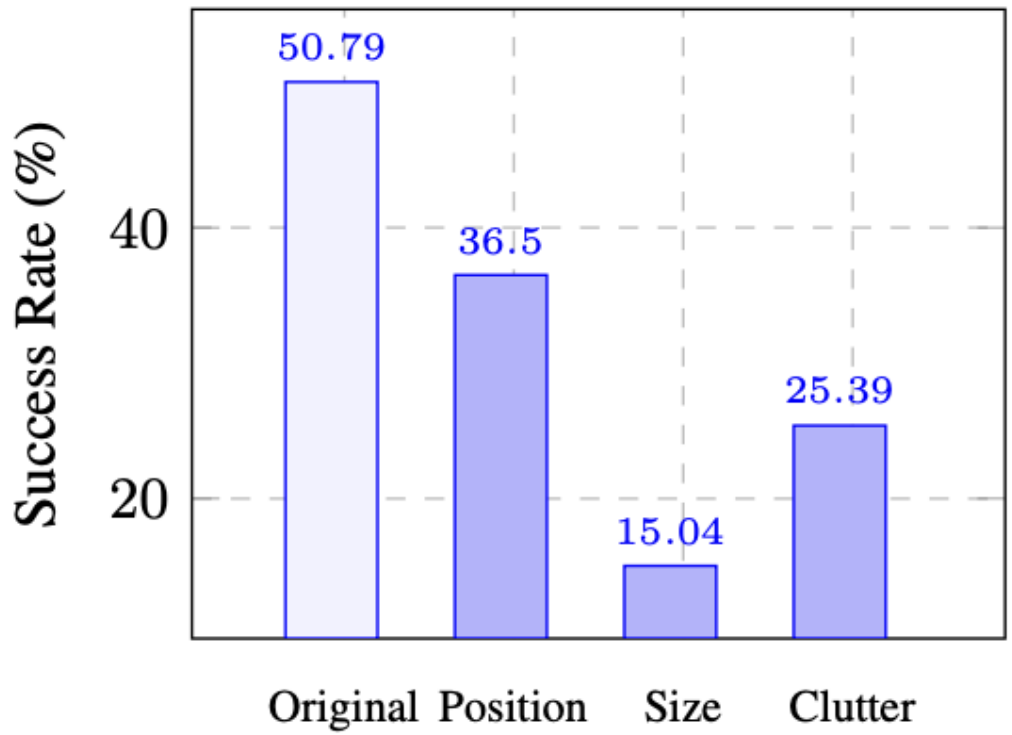
Related performance patterns, not identical success rates.

The result concerns this method and adapted task set. It is not a universal transfer guarantee.

Georgia Tech

The same goal fails after small layout changes

Original Figure 8: 28 selected previously successful tasks, not the full benchmark.



01 **Move the window**

The same task appears in a new position.

02 **Shrink the window**

Important controls become harder to access.

03 **Add clutter**

Unrelated windows distract the agent.

ROBUSTNESS **Shrinking the window: ~51% → ~15% on the selected task subset.**

Georgia Tech

Where do agents fail, even after choosing a sensible plan?

Failure patterns from the original analysis—and the challenge for longer workflows.

FAILURE 01

GUI grounding

The agent selects the wrong button or cell, or uses inaccurate click coordinates.

FAILURE 02

Tool knowledge

The model chooses an incorrect menu, command, or application procedure.

MULTI-APP WORKFLOWS

2.97%

GPT-4 + accessibility tree
Workflow-task success

WHY IT MATTERS **One missed step can break the whole task; recovery and verification matter.**

Georgia Tech

Who can verify success: the agent or the benchmark?

Execution-based checks are a strength, but an agent may lack the evaluator's information.

THE AGENT (WHILE WORKING)

Sees: screenshot / accessibility tree

Chooses actions and observes changes

Must judge success from the available interface.

≠

THE BENCHMARK EVALUATOR

Can inspect saved files and application states

Can compare results with expected outputs.

Applies task specific pass/fail rules.

How can an agent verify the result without the benchmark's answer key?

Georgia Tech

Strengths and limitations

What OSWorld contributes and what remains challenging.

STRENGTHS

01 **Executable tasks**

Agents act in working desktop applications.

02 **Cross-app coverage**

369 tasks include multi-application workflows.

03 **Outcome-based checks**

Measures actual files and computer state.

LIMITATIONS

01 **Finite coverage**

Cannot test every real-world task or application.

02 **Custom evaluation**

Tasks need setup and specific outcome checks.

03 **Safety blind spots**

Task success may miss harmful side effects.

A strong testbed for computer agents, but not a complete measure of real-world reliability.

Georgia Tech

Three conclusions and the question OSWorld 2.0 inherits

My interpretation of the OSWorld (2024) results.

01

RESULTS, NOT CLAIMS

A completed file or state matters more than a convincing explanation.

02

VISION IS NOT ENOUGH

GUI grounding, app knowledge, and input representation affect success.

03

VERIFY AND RECOVER

Longer workflows need agents that recognize and correct mistakes.

NEXT → OSWorld 2.0: Can stronger agents complete longer workflows?

Georgia Tech

Why OSWorld 2.0?

EXECUTABLE DESKTOP TASKS

12.24%

best agent on OSWorld, 2024 (humans 72.36%)

2024 diagnosis: GUI grounding, operational knowledge

IMPROVING PERFORMANCE

79–83%

frontier agents on OSWorld 1.0 tasks, 2026

Frontier agents now solve most original tasks

RELIABILITY GAP

Whole workflows?

short, narrow 1.0 tasks

2026 diagnosis: local actions fine, no task-level model

OSWORLD 2.0

20.6%

best agent, strict completion, 108 long workflows

new, harder tasks: not a regression

Can agents reliably complete realistic, long-horizon workflows, end to end?

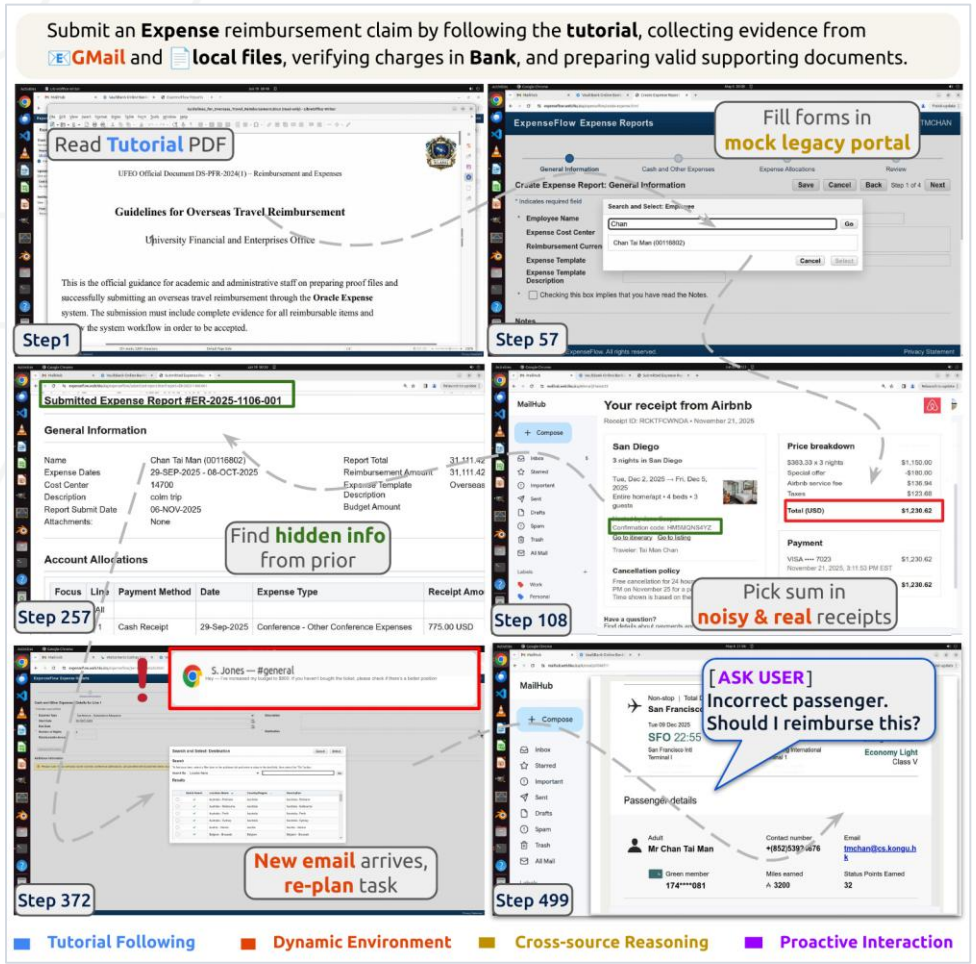
Different task sets and agent setups: the contrast motivates the question; it is not a controlled comparison.

Our view: short-task scores show local competence, not end-to-end reliability.

Sources: OSWorld (arXiv 2404.07972), Abstract, Table 5; OSWorld 2.0 (arXiv 2606.29537v2), §1, §3.2, Fig. 1 (right), Table 3. Diagram by presenter.

Georgia Tech

Task 008: an expense reimbursement



GOAL

Get the NeurIPS and Stanford trip costs reimbursed

INFORMATION SOURCES

- Policy guideline
- MailHub receipts
- VaultBank charges
- Prior ExpenseFlow report

DELIVERABLE

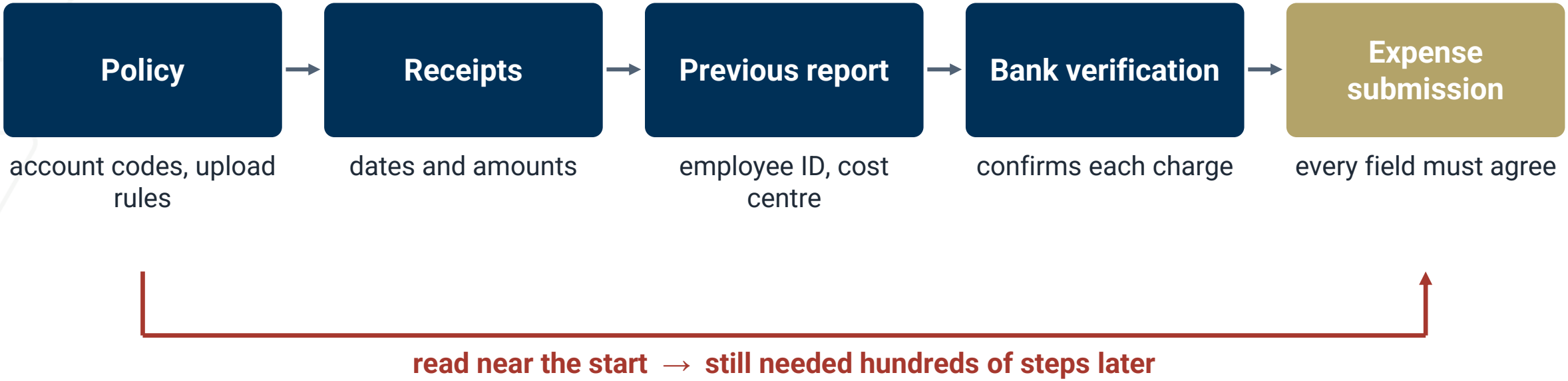
One correct, submitted ExpenseFlow report

The user states the goal, not the procedure.

Source: OSWorld 2.0 (arXiv 2606.29537v2), Fig. 1 (left; panels from a representative trajectory); App. H.1.1 (instruction, initial state).

Georgia Tech

Why is this workflow difficult?



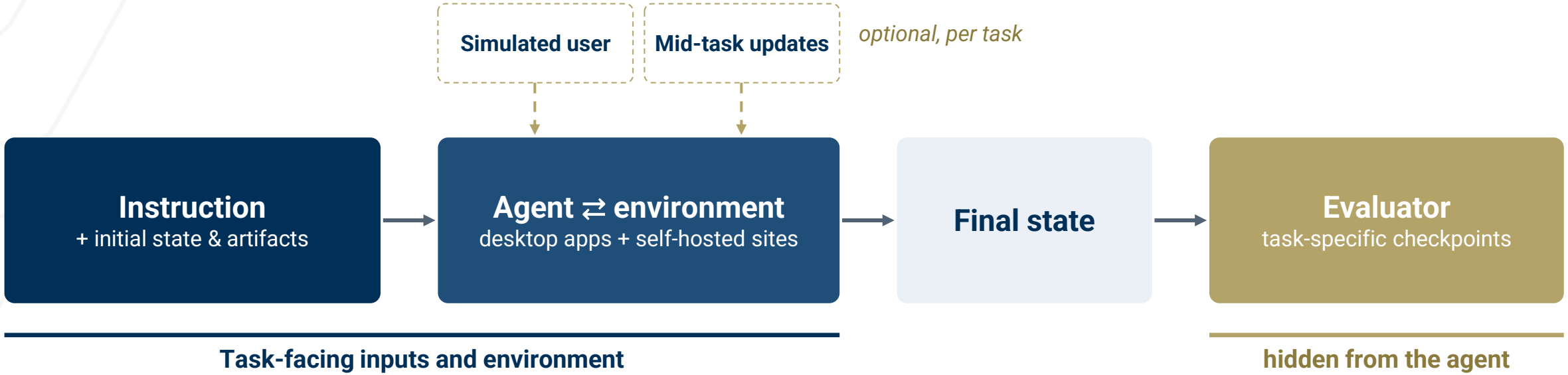
Correct individual actions do not guarantee a correct end-to-end workflow.

Our view: long-horizon dependencies make isolated GUI competence insufficient.

Source: OSWorld 2.0 (arXiv 2606.29537v2), App. H.1.1 (documented 493-step run; Analysis paragraph); §4.4. Chain simplified by presenter.

Georgia Tech

Anatomy of an OSWorld 2.0 task



108
tasks

31
self-hosted services

27.25
checkpoints per task (mean)

Source: OSWorld 2.0 (arXiv 2606.29537v2), §2.1, §2.1.2–2.1.3, §2.2, Table 5. Diagram by presenter.

Georgia Tech

Where do the tasks come from?



≈90% of final tasks

supporting channels, combined

Practitioner interviews

high quality · costly

Questionnaires

fast · few retained

LLM proposals

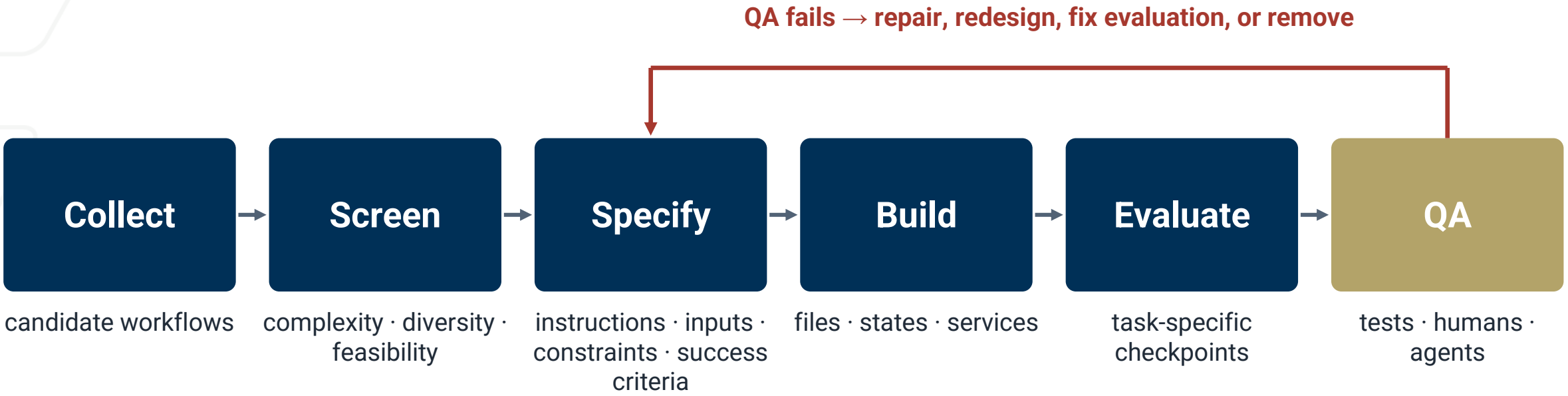
fast · unrealistic → ideas only

Realism and evaluability were hard to automate, so humans authored the tasks.

Source: OSWorld 2.0 (arXiv 2606.29537v2), §2.1.1, Fig. 2 (left). Only "≈90%" is reported; the 10% for the other channels combined is derived.

Georgia Tech

From task idea to executable benchmark

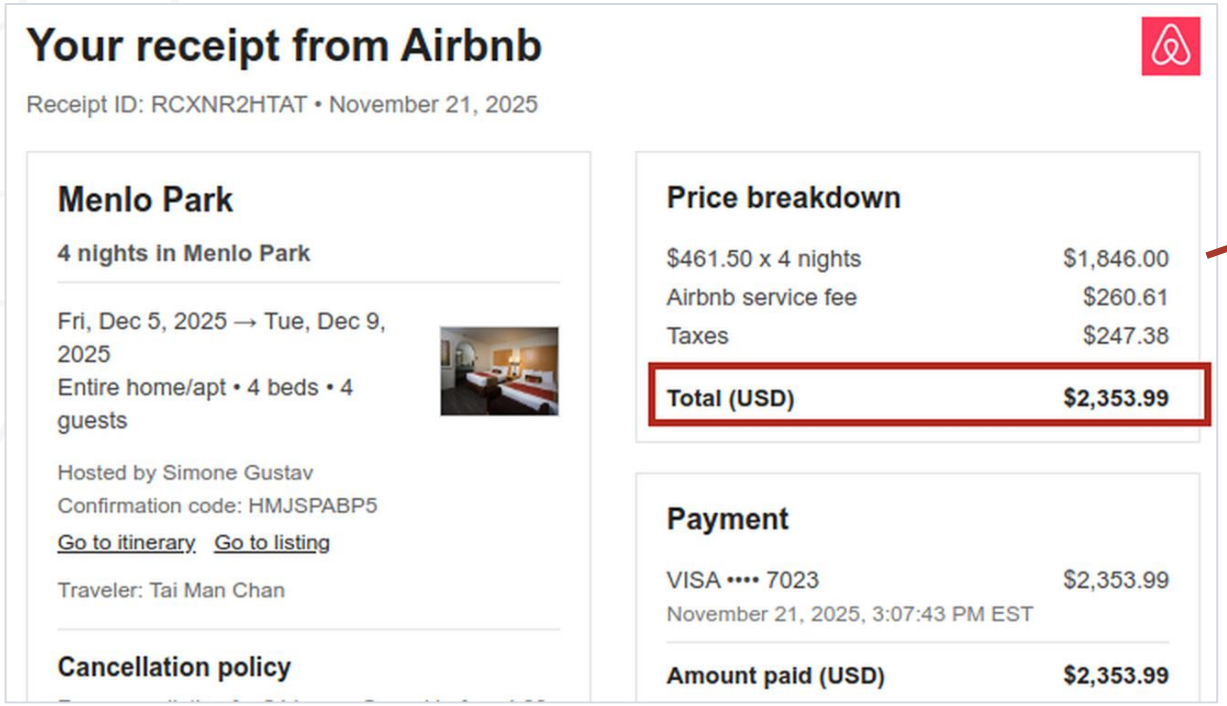


This is how humans build tasks, not how the tested agent runs.

Source: OSWorld 2.0 (arXiv 2606.29537v2), Fig. 2 and caption; §2.1, §2.1.1–2.1.4. Redrawn and simplified from Fig. 2 by presenter.

Georgia Tech

Realistic, consistent initial state



Your receipt from Airbnb

Receipt ID: RCXNR2HTAT • November 21, 2025

Menlo Park

4 nights in Menlo Park

Fri, Dec 5, 2025 → Tue, Dec 9, 2025
Entire home/apt • 4 beds • 4 guests

Hosted by Simone Gustav
Confirmation code: HMJSPABP5
Go to itinerary Go to listing

Traveler: Tai Man Chan

Cancellation policy

Price breakdown

$461.50 x 4 nights	$1,846.00
Airbnb service fee	$260.61
Taxes	$247.38
Total (USD)	**$2,353.99**

Payment

VISA •••• 7023	$2,353.99
November 21, 2025, 3:07:43 PM EST	
Amount paid (USD)	**$2,353.99**

MailHub: real Airbnb receipt (Task 008)

matches the receipt total

VaultBank · Travel transactions

Nov 21 · AIRBNB · −$2,353.99

Correct expense entry

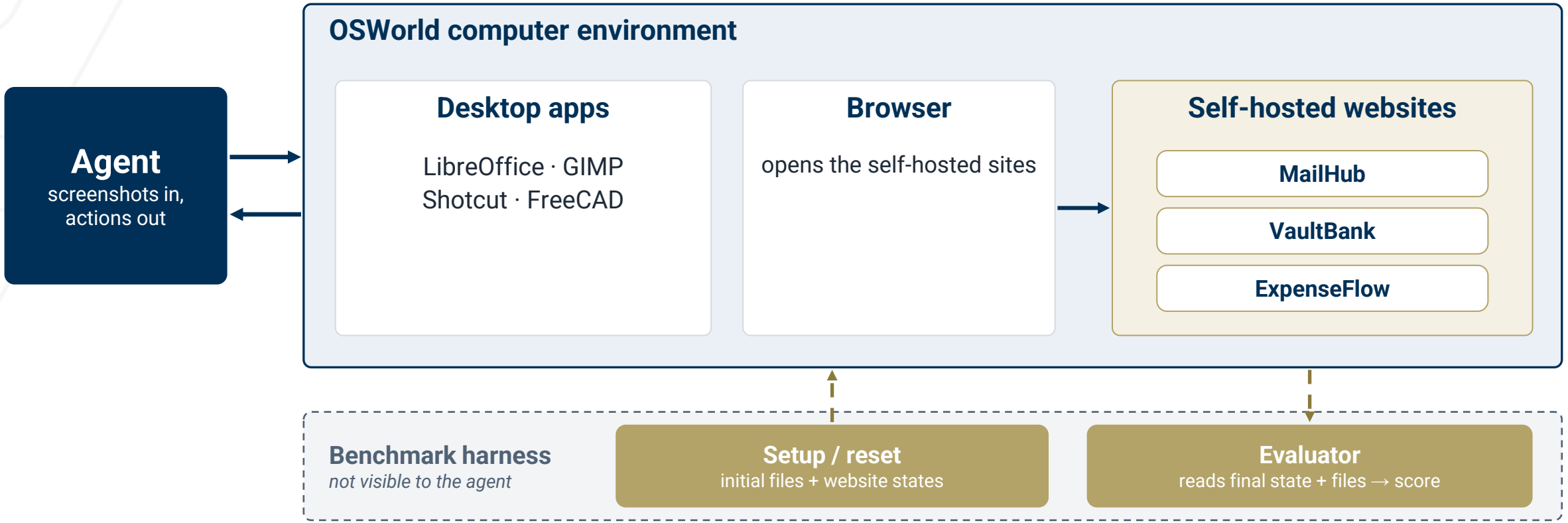
Employee ID and cost centre: found only in the prior report

Sources agree by construction, so cross-checking is both required and scoreable.

Source: OSWorld 2.0 (arXiv 2606.29537v2), §2.1.2; App. H.1.1 Figs. 20 (receipt, cropped; red box added), 21, 22 (bank row re-typed).

Georgia Tech

Self-hosted websites and real applications



Controlled data
no live-site drift or bot blocks

Repeatable resets
same initial state every run

Reproducible scoring
evaluator reads the final state

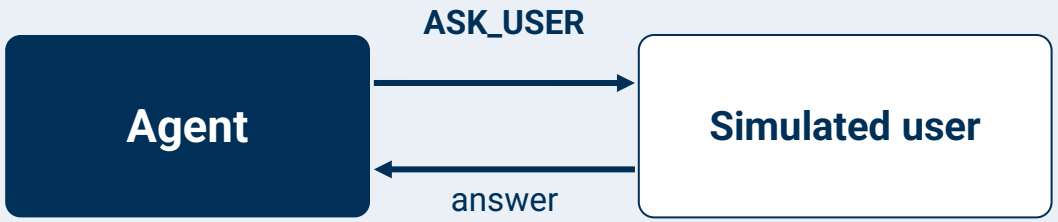
Source: OSWorld 2.0 (arXiv 2606.29537v2), §2.1.2; App. C.1–C.3. Diagram adapted from Fig. 13 by presenter.

Georgia Tech

Simulated users vs. dynamic environments

Simulated user

Agent PULLS information

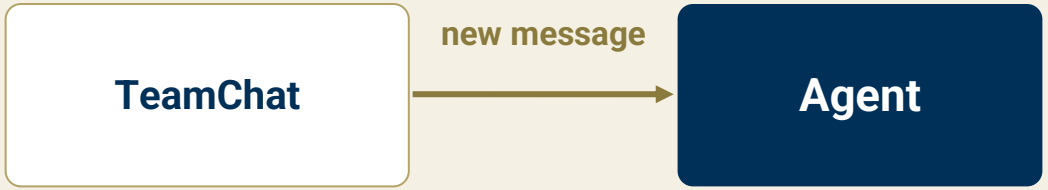


TASK 024

Funding certificate falls short → the agent asks for an updated one

Dynamic environment

Environment PUSHES updates



TASK 035

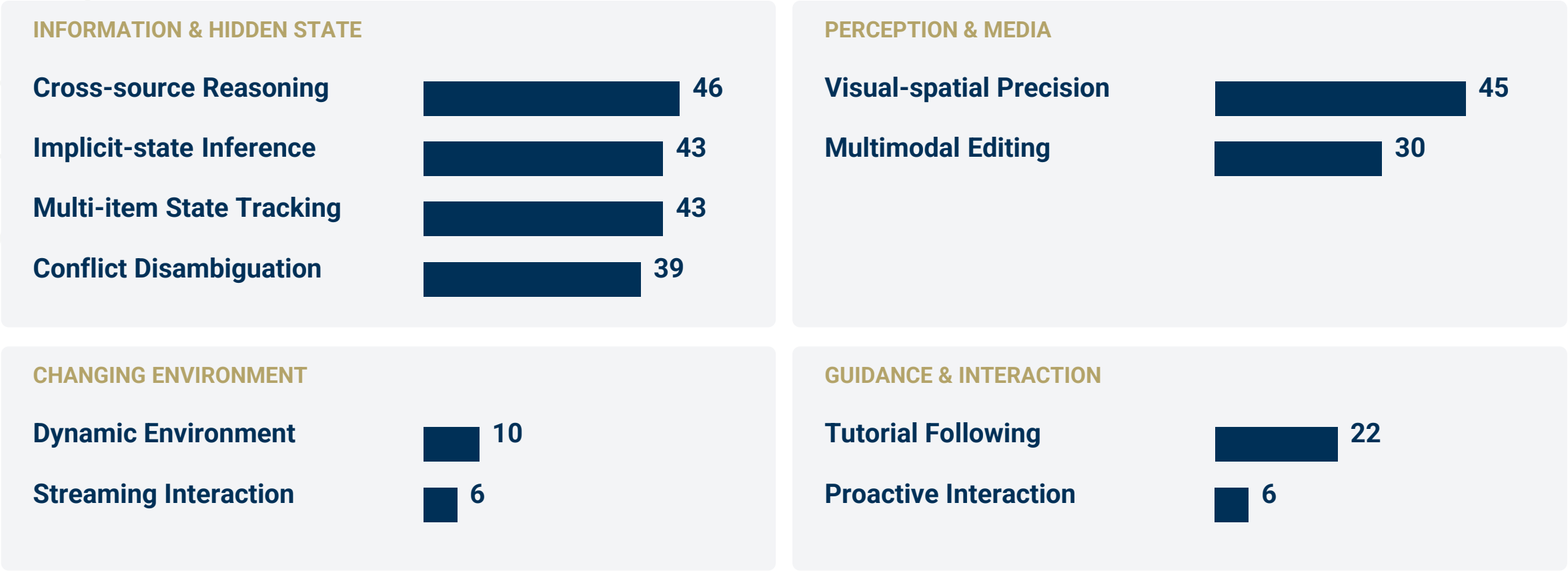
Manager's new message overrides an earlier purchase rule → the agent must revise

Source: OSWorld 2.0 (arXiv 2606.29537v2), §2.1.2; §3.1; App. E.1; H.2.2; H.2.3.

Georgia Tech

The ten challenge phenomena

Number of tasks (of 108) with each tag; one task can carry several



Our grouping of 290 overlapping tag assignments: 59% information and state, 26% perception.

Source: OSWorld 2.0 (arXiv 2606.29537v2), §2.2.2, Table 2 (counts). Grouping by presenter (cf. Abstract's “interaction-design” vs. “agent-pattern” challenges).

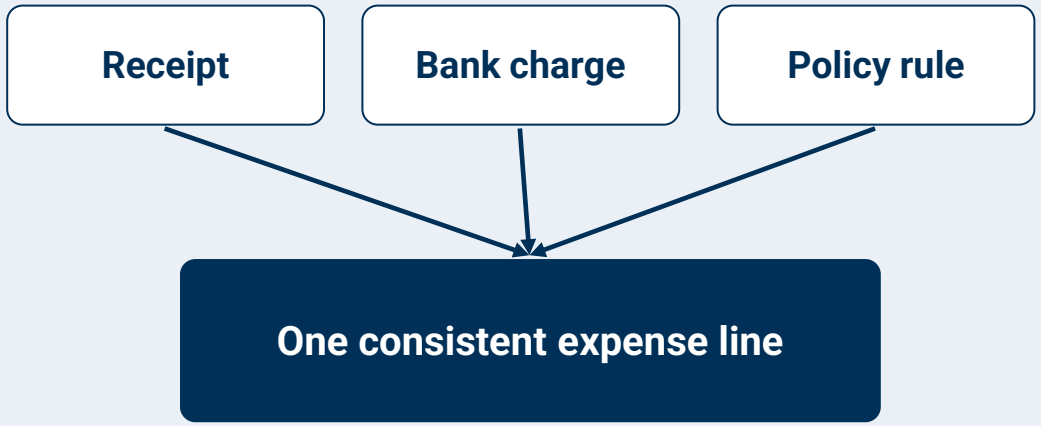
Georgia Tech

Cross-source reasoning vs. implicit-state inference

Both appear in Task 008

Cross-source reasoning

- Facts appear in several sources
- Agent must reconcile them



Implicit-state inference

- Required information isn't given explicitly
- Agent must infer what is missing and where to find it



Source: OSWorld 2.0 (arXiv 2606.29537v2), Table 2/10 and App. C.5 (definitions); App. H.1.1 (instruction; steps 9, 102, 139, 249). Diagrams by presenter.

Georgia Tech

Multi-item tracking vs. conflict disambiguation

Task 035: fill purchase orders while TeamChat rules change

new messages arrive mid-task



Initial rules
channel post + DMs

Exception
one rejected item approved

Correction
budget cap and vendor changed

Final PO decisions
must reflect the latest rules

Multi-item state tracking

Keep every item's status correct

Conflict disambiguation

Decide which instruction is authoritative

Source: OSWorld 2.0 (arXiv 2606.29537v2), App. H.2.2 (Figs. 36–38); Table 2/10; App. C.5. Timeline simplified by presenter.

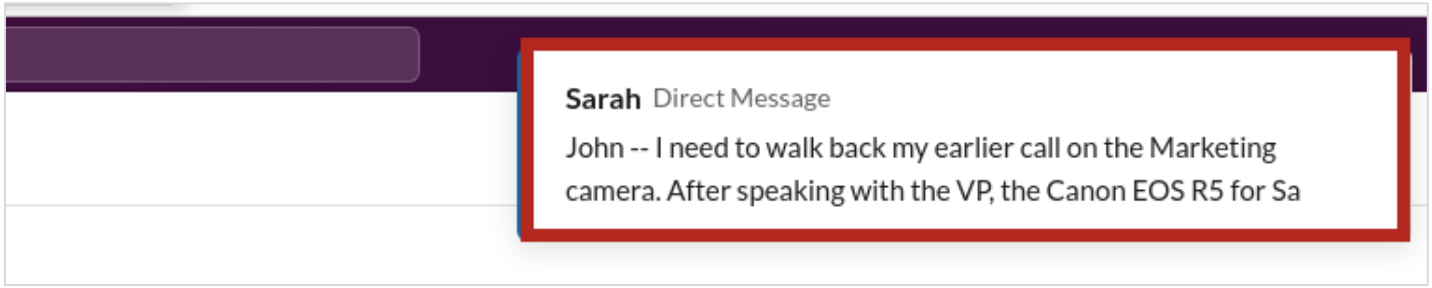
Georgia Tech

Dynamic environment vs. streaming interaction

Dynamic environment

task information changes

fails as: stale rules

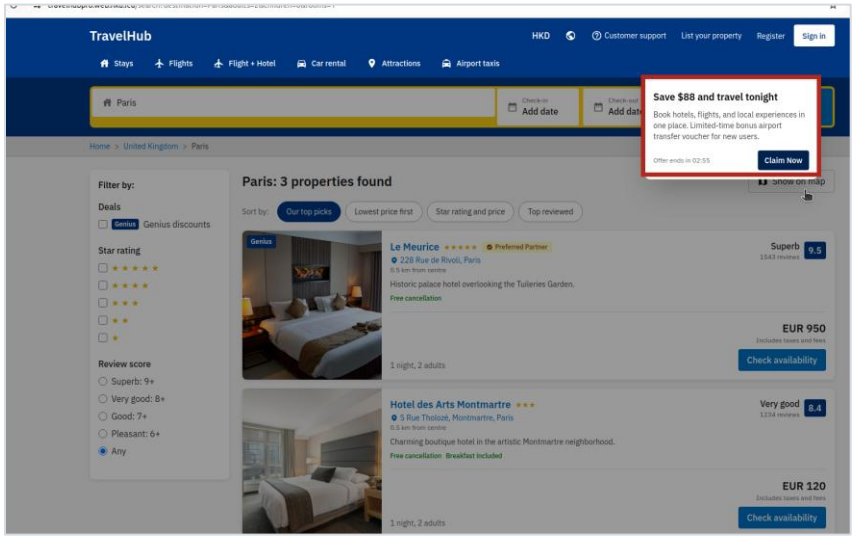


Task 035: a manager's message arrives mid-task and overrides an earlier decision

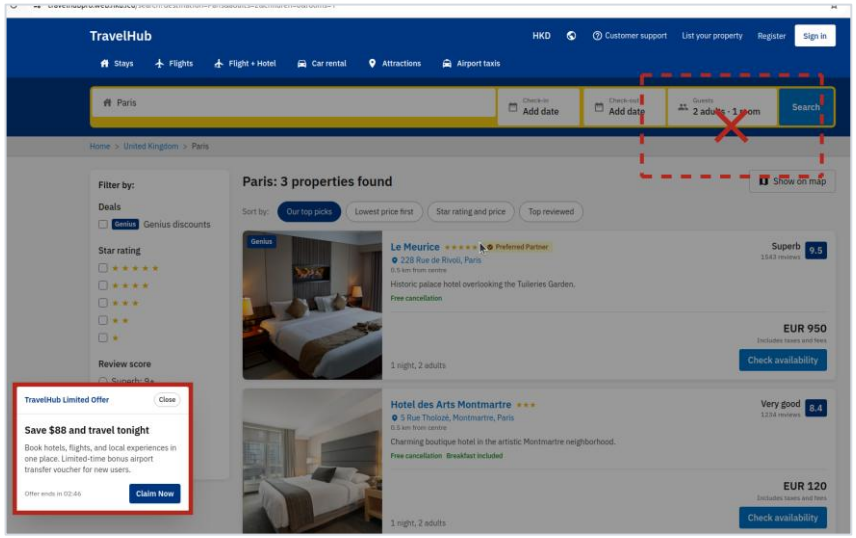
Streaming interaction

screen changes between look and click

fails as: stale click



1 Observe: pop-up here



2 Click lands at the old spot X; pop-up has moved

Source: OSWorld 2.0 (arXiv 2606.29537v2), §2.1.2; App. C.5; Fig. 12 and App. H.2.1–H.2.2 (screenshots cropped; red marks added).

Georgia Tech

Tutorial following vs. proactive interaction

Tutorial following

Read external guidance → adapt it to this task

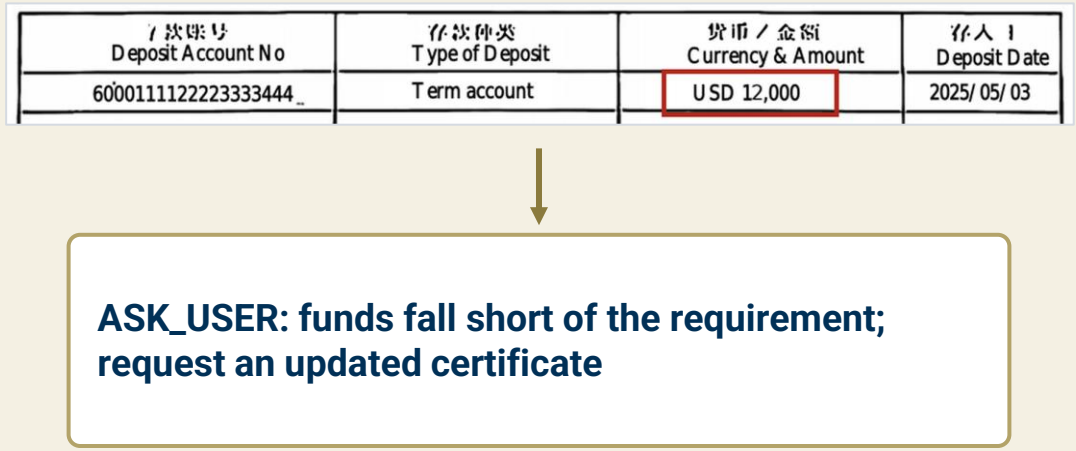
PDF/web guides · video tutorials · prior completed work

- Conference - Accommodation: 10110
- Conference - Airfare (Basic Economy Class): 10120
- Conference - Airfare (Other than Basic Economy Class): 10120
- Conference - Other Conference Expenses: 10130
- Conference - Subsistence Allowance: 10148

Task 008: the guideline maps each expense type to an account code

Proactive interaction

Notice missing evidence → ask, don't guess



Task 024: USD 12,000 on file vs. $18,000 needed (ASK_USER box is a schematic)

Source: OSWorld 2.0 (arXiv 2606.29537v2), Table 2/10; App. C.5; H.1.1 (Fig. 17, cropped); H.2.3 (Fig. 39, cropped; red box added); H.3.1–H.3.3.

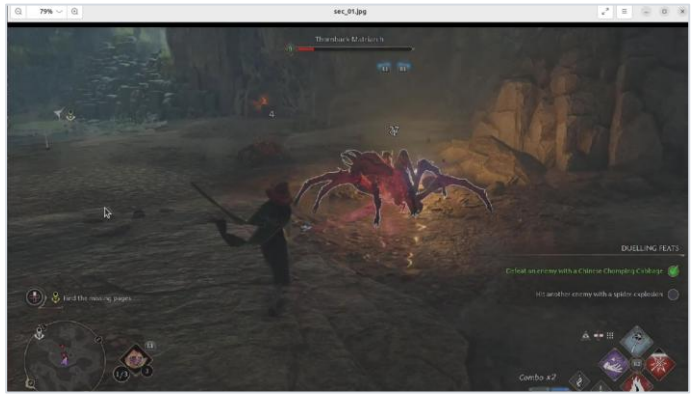
Georgia Tech

Multimodal editing vs. visual-spatial precision

Multimodal editing

create or modify non-text media

Task 053: mask every spider



input frame (Fig. 43)



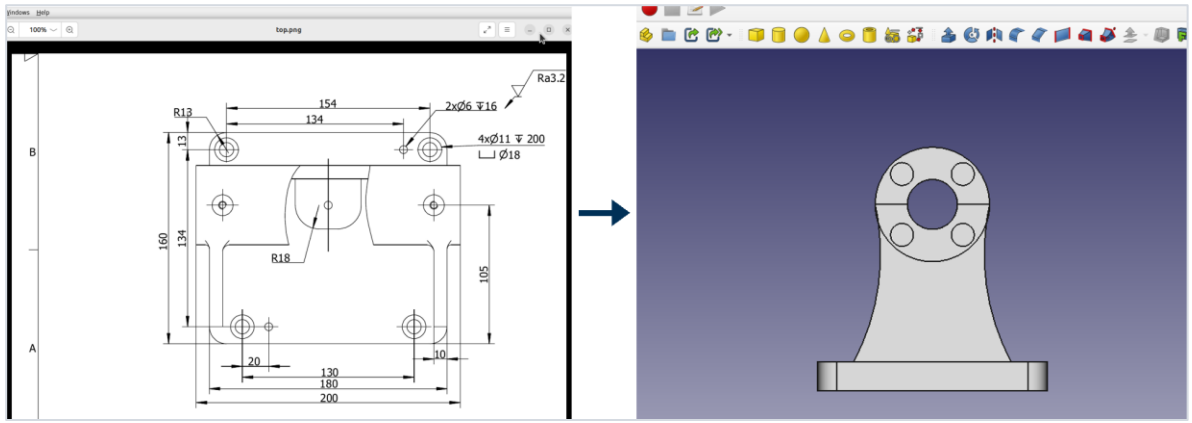
masked output, sampled frames (Fig. 46)

Paper: one box misplaced (≈8.5 s), others oversized → partial 0.69

Visual-spatial precision

exact geometry, position or timing

Task 103: rebuild from a drawing



drawing (Fig. 28) → agent's model (Fig. 32): cylinder and slot off → partial 0.35

Source: OSWorld 2.0 (arXiv 2606.29537v2), Table 2/10; App. C.5; H.2.4 (Figs. 43, 46; crops); H.1.2 (Figs. 28, 32; composited by presenter).

Georgia Tech

Binary completion vs. partial rewards

ILLUSTRATIVE: 4 equally weighted checkpoints

- ✓ Correct report header
- ✓ Correct expenses
- ✗ **Correct attachments**
- ✓ Submitted successfully

Partial score

3 / 4 = 75%

Binary completion

0

Real Task 008 run: partial 0.76, binary 0

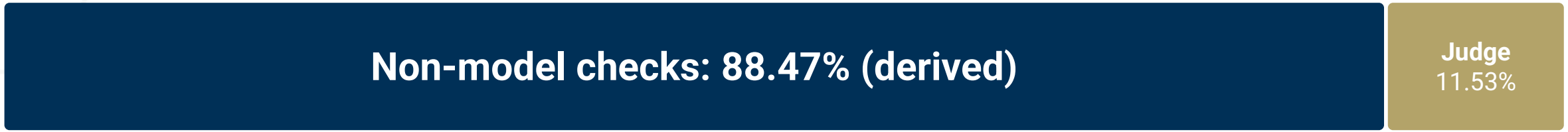
Partial credit shows progress; the last exact requirements decide completion.

Source: OSWorld 2.0 (arXiv 2606.29537v2), §2.1.3 (checkpoints, order-free); App. G.2 (binary = partial score of 1.00); App. H.1.1 (0.76). Toy example invented.

Georgia Tech

Functional evaluation vs. model-based judging

Share of the total score



Functional checks

- Final website state + uploaded files
- Output files and artifacts
- Fuzzy checks where several outputs are valid

Model judge

- Open-ended items (e.g. email tone)
- Objective yes/no checklist
- Validated on near-miss wrong states

No task relies on the model judge for more than half of its score.

Source: OSWorld 2.0 (arXiv 2606.29537v2), §2.1.3 and Table 5 (11.53% reported; 50% cap); §2.1.4; App. C.2. 88.47% = 100% − 11.53% (derived, not stated in the paper).

Georgia Tech

Quality assurance: can we trust the benchmark?

1 Auto unit tests
rubric implemented as code

→

2 Two human solves
solvable? rubric right?

→

3 Agent rollouts
unexpected solution paths

Reward-hacking audit
credit without doing the task

False-negative audit
correct work under-scored

Disagreement → revise or remove the task

What can the benchmark verify that the agent cannot?

Privileged evaluator

- Backend state + expected checkpoints
- Can check results against ground truth

Working agent

- Screenshots + accessible evidence
- Must independently verify its work

A reliable benchmark score doesn't mean the agent knows it succeeded.

Our view: a harder benchmark helps only if scoring tells real completion from shortcuts and near-misses.

Source: OSWorld 2.0 (arXiv 2606.29537v2), §2.1.4; Fig. 2 (right); App. E.1; G.3 (Task 024); H.4.2 (Task 052); §6; App. C.2 (evaluator); H.1.1 (Task 008). Diagram by presenter.

Georgia Tech

OSWorld -> OSWorld2.0

	OSWorld 1.0	OSWorld 2.0
Task horizon (avg. agent steps)	<30	>250
Cross-app tasks	Supported (minority)	Majority (≥2 apps/services, info-dependent)
Self-hosted web environments	—	31 websites
Input artifact source	Mixed/synthetic	Authentic
Challenge phenomena	—	10 annotated tags
Scoring	Binary	Partial reward (avg. 27.25 ckpts)
Model-based evaluation	—	11.53% of score
Safety audit	—	8 diagnostic checks
User interaction	—	Simulated user

	OSWorld (2024)	OSWorld 2.0 (2026)
Task set	369 tasks	108 longer workflows
Cross-app tasks	Supported but minority	Majority (>= 2 apps/services, info-dependent)
Workspace	Desktop apps and web tasks	31 self-hosted sites, richer workspace state
Task Horizon (avg. agent steps)	< 30	> 250
Interaction	Task instruction and environment	Simulated user and mid-task updates
Scoring	Primarily binary success	Binary completion + partial checkpoints
Diagnostics	Application and workflow results	10 challenge tags + separate safety audit

Georgia Tech

Task horizons grow by about 48×

~2 minutes

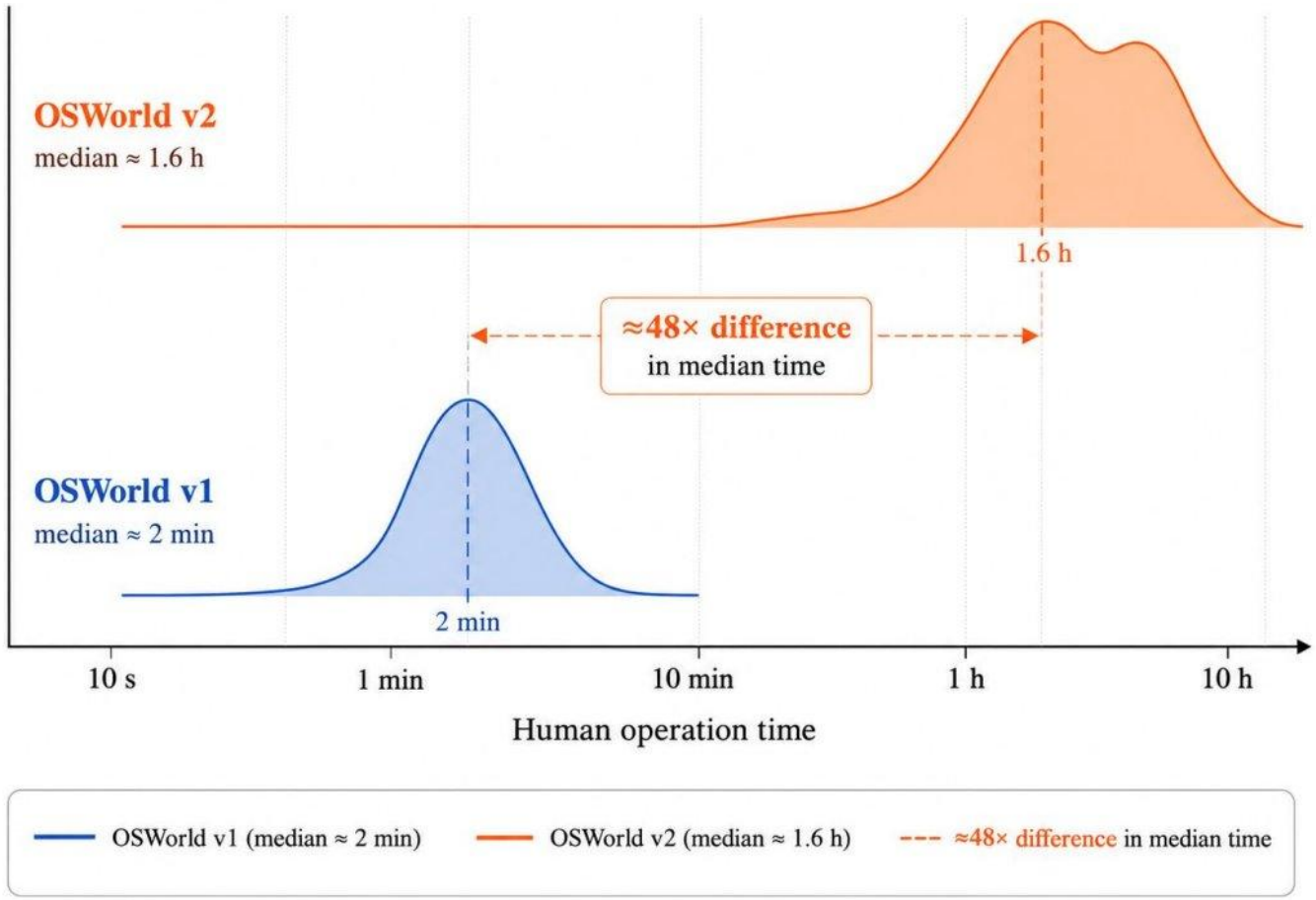
OSWorld median

~1.6 hours

OSWorld 2.0 median

69.6% of 2.0 tasks

exceed one human hour



Source: OSWorld 2.0, Fig. 3 and §2.2.1; OSWorld, Fig. 4

Georgia Tech

Broader workflows across professional domains

108 tasks across 7 domains

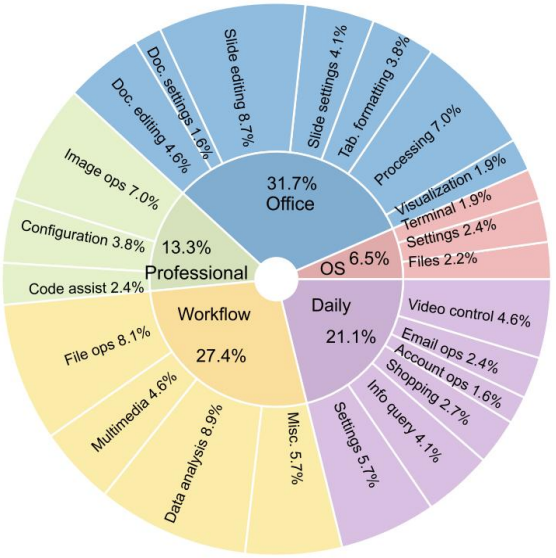
and 21 subcategories

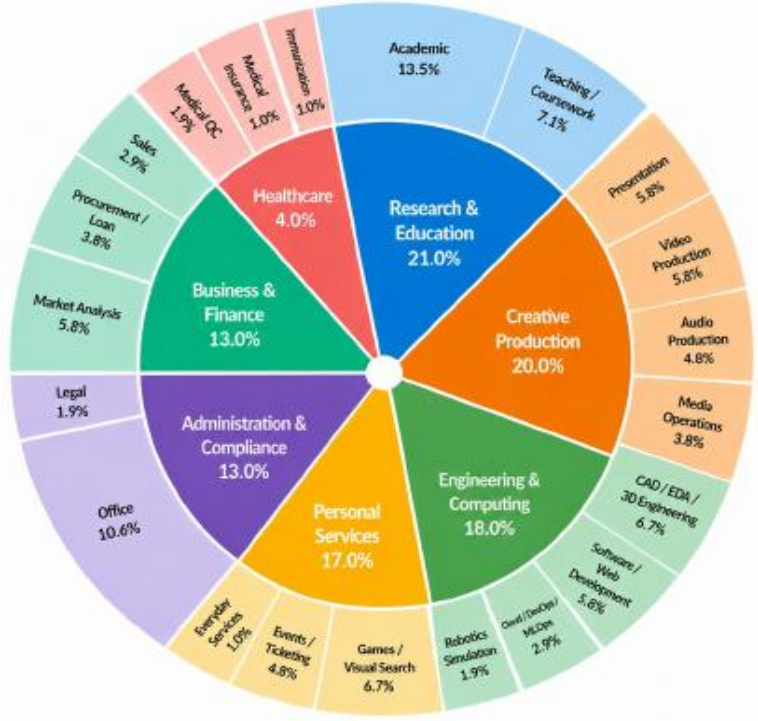
Research & Education +

Creative Production: over 40%

64.8% of tasks require

at least 2 apps or services





Georgia Tech

Experimental setup and scoring

7 model families : Claude Opus 4.8, Claude Opus 4.7, Claude Sonnet 4.6, GPT-5.5, Qwen 3.7-Plus, MiniMax M3, and Kimi 2.6.

- **Screenshot only observations**
- Claude models act through the native claude_computer_use tool, while the remaining models emit pyautogui code actions.
- **Step budget : 500.**

- Two tool use settings : Batched (multiple tool calls per step) and single (one call per step) GPT-5.5 always batches its tool calls, whereas Claude models batch only when the batch tool is explicitly enabled.
- A 3 second pause is inserted after each action.

Georgia Tech

The strongest setting completes only 20.6%

	Model	Binary (%)	Partial (%)	Cost/task	Tool calls/task	Out tok/task	Steps/task
Batched actions	Claude Opus 4.8	**20.6**	**54.8**	~$72.4	481.8	224K	103
	Claude Opus 4.7	18.2	48.91	~$33.6	597.1	150K	160.7
	GPT-5.5	13.0	49.5	~$25.5	149.8	37.1K	95.2
Single action	Claude Opus 4.8	18.5	49.3	~$76.1	190.5	259.5K	190.5
	Claude Opus 4.7	13.9	49.1	~$35.8	318.4	150.5K	318.4
	Claude Sonnet 4.6	8.3	41.5	~$22.3	253.3	185.9K	253.3
	MiniMax M3	4.6	22.3	~$2.4	326.7	70.8K	326.7
	Kimi 2.6	4.6	22.1	~$6.6	179.3	63.0K	179.3
	Qwen 3.7-Plus	2.8	21.5	~$3.8	173.5	28.9K	173.5

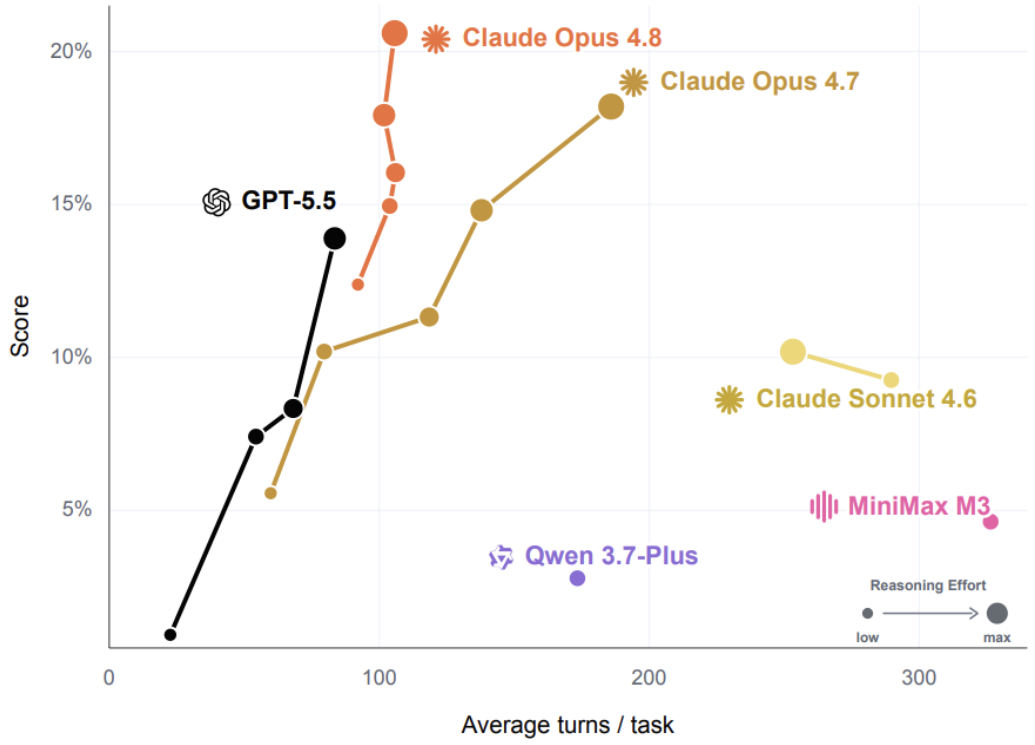
Model / tool setting	Binary %	Partial %	Output tokens / task
Opus 4.8 / batched	**20.6**	**54.8**	**224K**
Opus 4.7 / batched	18.2	48.9	150K
GPT-5.5 / batched	13.0	49.5	37.1K
Opus 4.8 / single	18.5	49.3	259.5K
Opus 4.7 / single	13.9	49.1	150.5K
Sonnet 4.6 / single	8.3	41.5	185.9K
MiniMax M3 / single	4.6	22.3	70.8K
Kimi 2.6 / single	4.6	22.1	63.0K
Qwen 3.7-Plus / single	2.8	21.5	28.9K

Georgia Tech

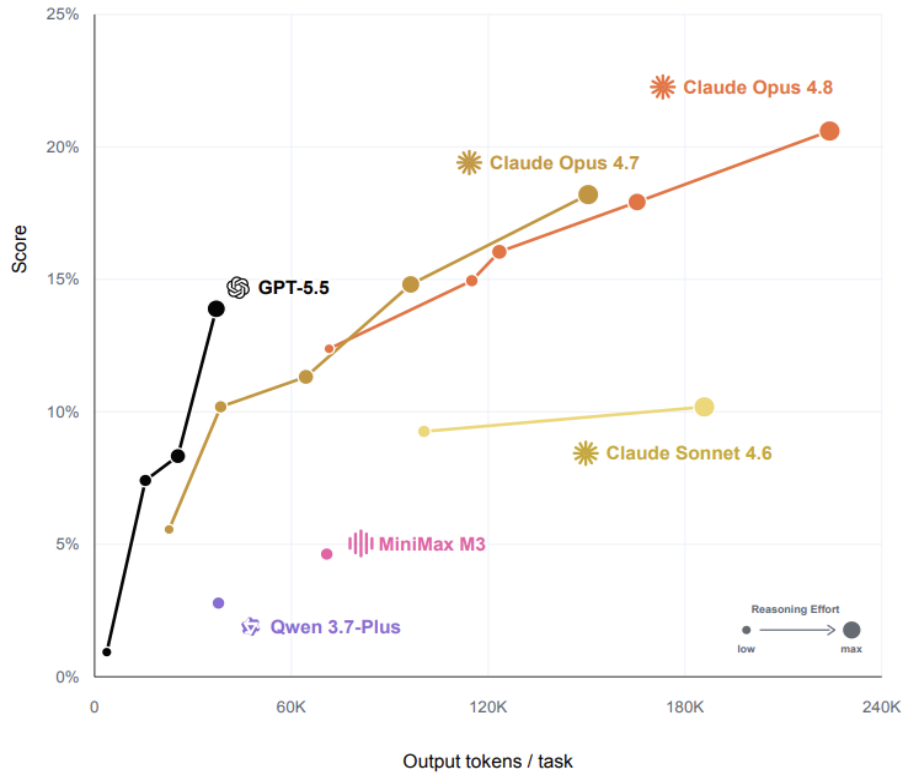
Results : Binary Completion

Higher completion comes with more output tokens

Model	Tokens	Success
GPT-5.5	37.1K	13.0%
Opus 4.7	150K	18.2%
Opus 4.8	224K	20.6%



Binary completion against average turns per task



Binary completion against Output Tokens/task

Georgia Tech

Binary completion rate collapses as the task horizon grows

Under 45 minutes:

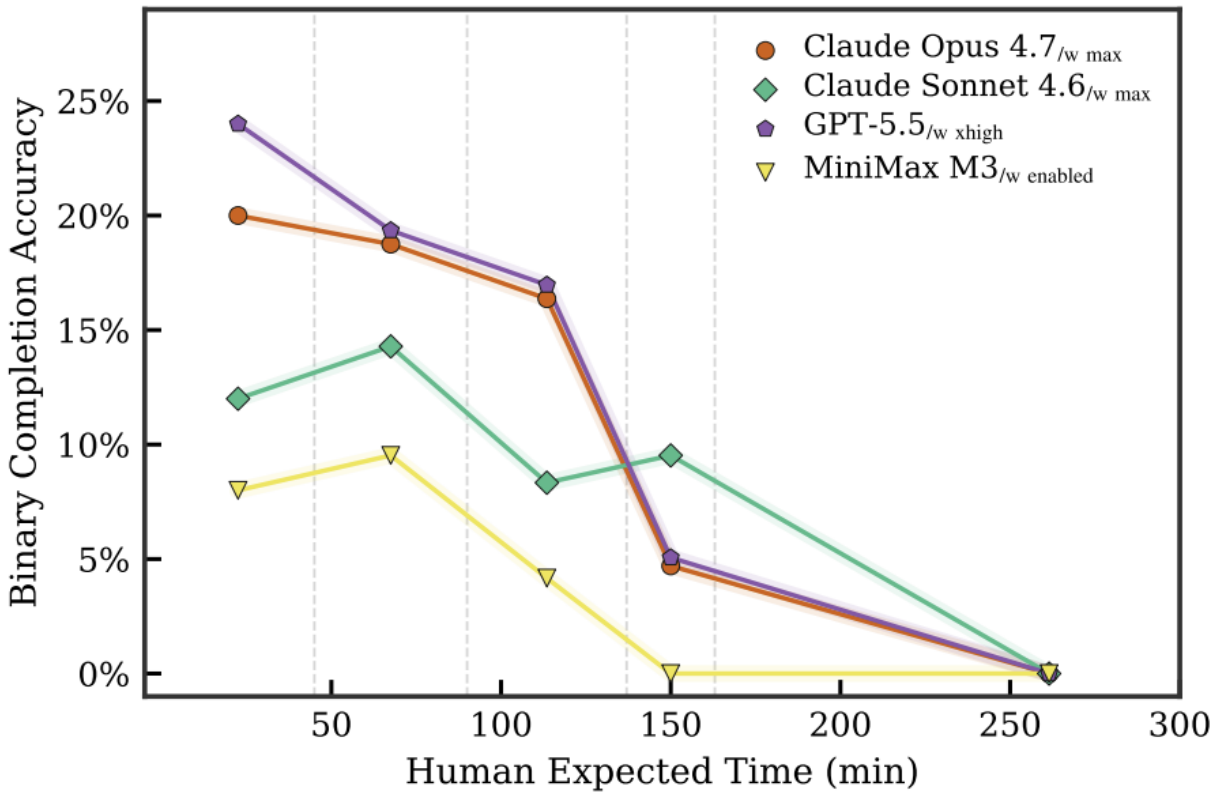
strongest plotted models reach 20–24%

137–163 minutes:

no plotted model exceeds 10%

Above 163 minutes:

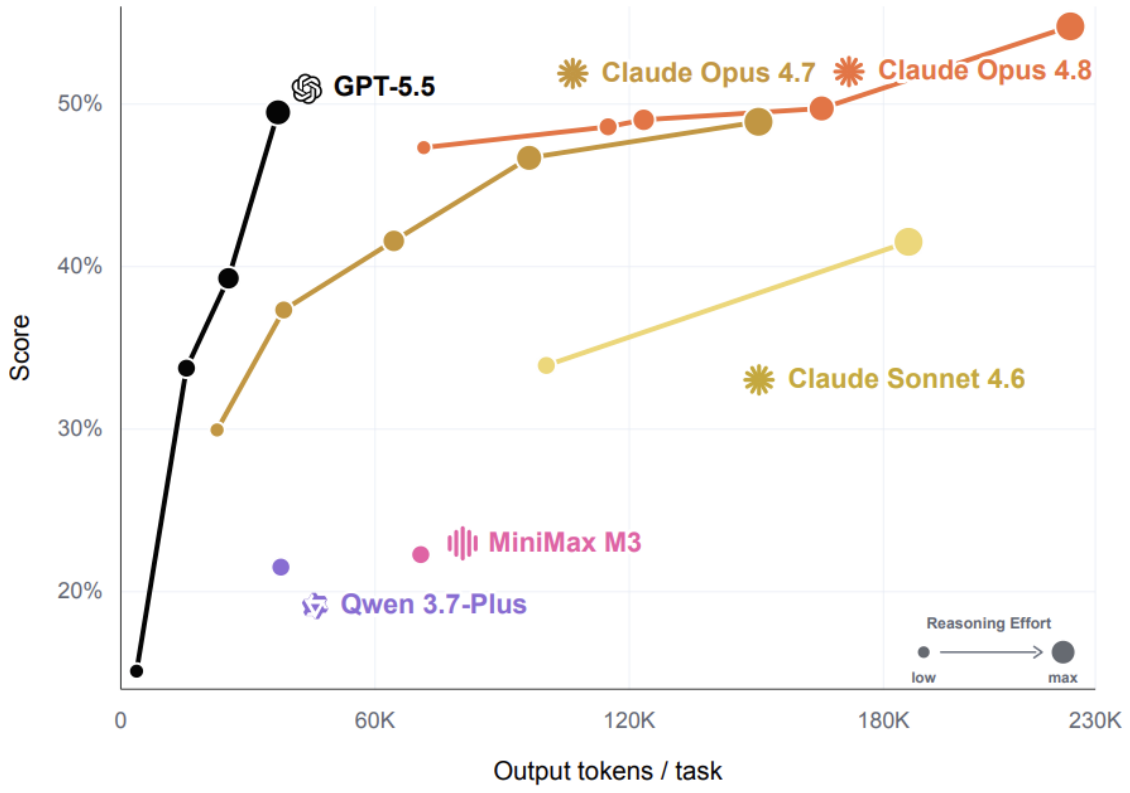
0% in the retained plotted models



Binary completion accuracy by human-annotated expected task time

Georgia Tech

Partial progress and full completion tell different stories



Partial reward against average output tokens per task

Batched model	Partial score	Binary completion
GPT-5.5	49.5%	13.0%
Opus 4.7	48.9%	18.2%
Opus 4.8	54.8%	20.6%

- GPT-5.5 and Opus 4.7 earn similar partial scores
- Opus 4.7 converts that progress into more completed tasks
- The remaining difficulty includes exactness and final verification

Georgia Tech

Recent agents: progress needs comparable scores

Model	Partial reward	Binary completion	Evaluation context
GPT-6 Astra	73.5%	Not reported here	Offline v2026.08.08
GPT-6.1 Sol	71.4%	Not reported here	Offline v2026.08.08
Claude Fable 5.1	77.9%	41.7%	August release; modified eval

Sol approaches Astra at lower cost

Maximum effort: Sol approaches Astra's partial score at 1/7th cost per task

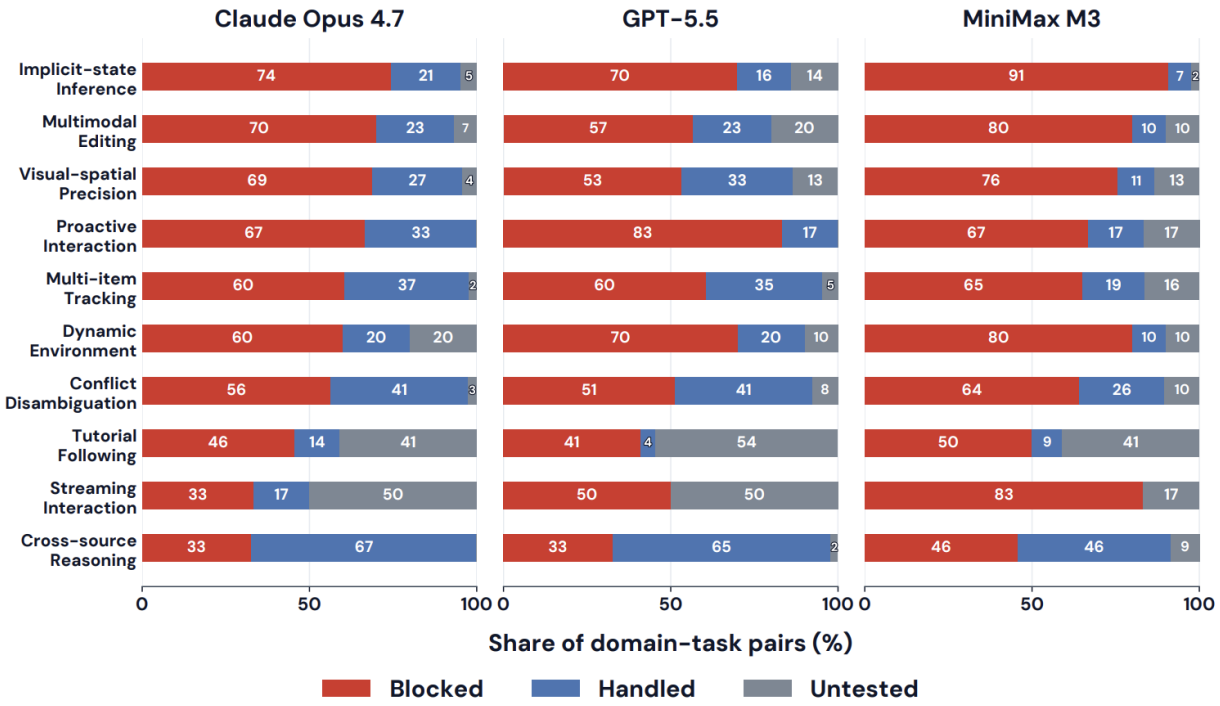
These are vendor-reported results, not a controlled cross-vendor comparison. Differences in task release, evaluation configuration, safeguards, and scoring prevent a definitive ranking between Fable and the OpenAI models.

Sources: OpenAI, GPT-6.1 Sol (Sep 29, 2026); Anthropic, Fable 5.1 (Sep 1, 2026)

Georgia Tech

Why agents Fail?

Exposure attribution across ten challenge phenomena



Label	Criterion
Handled	Agent reaches the challenge and handles it, including task-consistent workarounds.
Blocked	Agent reaches the challenge and fails because of it.
Untested	Agent never reaches the challenge, fails for an unrelated reason, or shortcuts past it.

All current agents are weak at recovering and maintaining hidden state

Visual versus interaction capability separates the two frontier models.

Selected partial scores	GPT-5.5	Opus 4.7
Visual-spatial precision	51.2%	43.9%
Multimodal editing	47.0%	44.0%
Proactive interaction	43.1%	52.0%

Georgia Tech

What does OSWorld 2.0 test about vision?

Vision is central to the interface, but the overall score cannot isolate visual capability.

Visual demand	What the agent must do	What the result tells us
Screen reading / grounding	Read labels and locate controls	Can it access and operate the interface?
Spatial / media understanding	Check geometry, layout or timing	Can it interpret and verify visual content?
Reasoning over screen content	Reconcile records and track updates	Vision carries evidence; memory also matters

45/108 visual-spatial tasks **30/108 multimodal editing** **6/108 streaming**

Tags overlap. These counts do not measure the share of tasks that require vision.

Takeaway: screenshot-only observations do not isolate vision's contribution.
A matched screenshot / structured-text / combined ablation would test necessity.

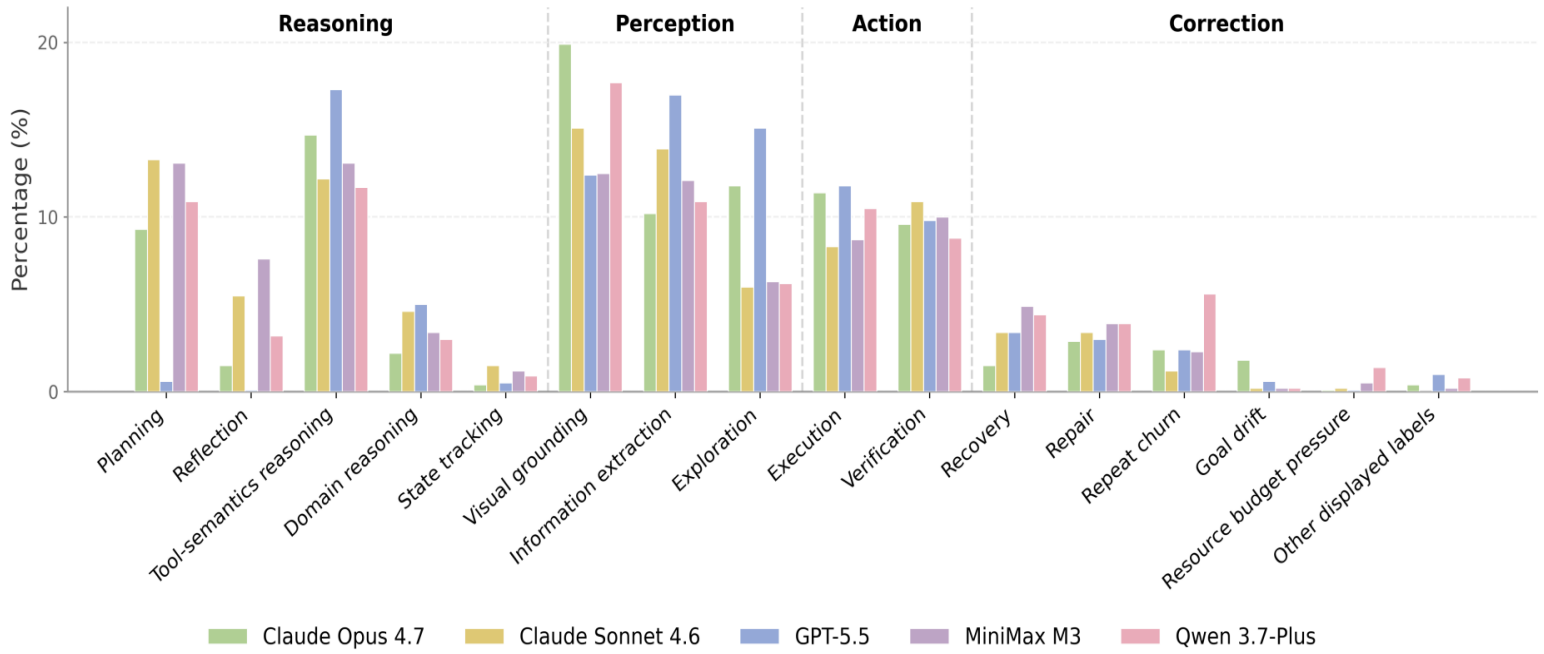
Sources: OSWorld 2.0, §3.1, Table 2, App. G.3; OSWorld, Table 5

Georgia Tech

Action Pattern :

Agents allocate little activity to recovery and repair

GUI clicks, terminal commands, keyboard shortcuts, and waiting account for 80% of all actions



Visual grounding:
15.5%

Tool-semantics reasoning:
13.8%

Information extraction:
12.8%

Execution:
10.1%

Recovery and repair
< 7%

Georgia Tech

5 Recurring Qualitative Failure Modes

Failure mode	Example or consequence in the paper
Information grounding	Purchase orders miss a later TeamChat correction
Perception-action timing	A moving pop-up makes the planned click stale
Domain knowledge	A plausible FreeCAD model has incorrect geometry
Verification and reflection	A submitted reimbursement still has incorrect details
Long-horizon state drift	Earlier evidence disappears from the working context

Georgia Tech

Limitations and Future work

- **Coverage remains incomplete**

  108 workflows cannot represent every profession or task type

- **Expansion is expensive**

  Realistic artifacts, hosting, scoring, and audits require substantial work

- **Scores depend on the benchmark**

  Task mix, stochastic runs, and benchmark-specific exploits affect interpretation

- **Completion and safe execution are separate**

  Safety audits report harmful side effects and interface bypasses

Georgia Tech

Discussion Questions

Real computer interaction persists.

Reliable completion over long workflows remains difficult.

- How should we balance full completion, partial progress, and safe execution?

- What would help most: persistent task memory, better verification, or faster perception-action loops?

- How can a benchmark reward valid shortcuts while detecting exploits?

Sources: OSWorld, §§3–4; OSWorld 2.0, §§3–4 and §6