# MKA: Evaluating Multimodal Knowledge Agents on end-to-end professional tasks

**Anima Singh**[*], **Suhas Yogin**[*], **Weicheng Kuo**[*], **Ivor Rendulic**, **Ivan Jurin**, **Antoine Yang**, **Oran Kelly**, **Rohan Doshi**, Hammad Ayyubi, Ioana Bica, Sergi Caelles, Matthew Conniff, Dan Graur, Anastasija Ilić, Ahmet Iscen, Chao Jia, Tomasz Kępa, Chunyuan Li, Noah Ó Donnaile, Filip Pavetic, Goran Petrovic, Khalid Salama, Zhicheng Wang, Polina Zablotskaia, Yiqian Zhang, Ming Zhao, **Benjamin Caine**, **Mario Lučić**, Melvin Johnson, Radu Soricut and Jean-Baptiste Alayrac
Google DeepMind

**We introduce the Multimodal Knowledge Agent (MKA) benchmark, which measures how well autonomous agents perform end-to-end professional knowledge work that depends on visual understanding. Each task requires an agent to extract and combine information from multiple visually dense files, such as CAD blueprints, electronics schematics, radiographs, complex charts and geospatial maps, and to produce a standalone deliverable in `.xlsx`, `.pptx`, `.pdf`, `.docx`, `.html` or `.png` format. Existing benchmarks leave this capability largely untested: removing image-viewing tools leaves Gemini 4 Carbon's score on GDPval (Patwardhan et al., 2025) unchanged (Section 4.3), and single-format evaluations such as GDP.pdf (Garre et al., 2026) and GDP.xlsx (Surge AI, 2026) do not require reasoning across files or modalities. MKA contains 200 tasks across 10 economic sectors. Deliverables are graded against expert-weighted rubrics by a sandboxed agentic autorater that agrees with human domain experts on 96.2% of criterion judgments in our expert-graded study (Cohen's $\kappa = 0.91$), scored by weighted satisfaction rate (WSR) overall and on frontier criteria unsaturated by prior-generation models (Climb-WSR). Across the frontier models we evaluate (Gemini 4 Carbon, Claude Fable 5.2, Claude Opus 5.5, and GPT-6 Astra), Claude Fable 5.2 (56.6% Climb-WSR, 73.3% Overall WSR) and Gemini 4 Carbon (54.3% Climb-WSR, 71.9% Overall WSR) achieve the highest scores, while Carbon uses 5.7× fewer tokens. Even for the strongest models, Climb-WSR remains below 57%, leaving substantial headroom for multimodal agents.**

## 1. Introduction

Real-world knowledge work is deeply visual. Professionals routinely read technical diagrams and schematics, interpret complex charts (Wang et al., 2024c), and work with messy scanned documents (Ma et al., 2024a)—all while visually verifying the formatting and layout of their own work products. Yet today's frontier agents, shaped by coding-centric training (Jimenez et al., 2024; Merrill et al., 2026; Wang et al., 2024a), exhibit a surprising degree of text-and-code bias rather than using vision as a tool. Given a workspace of files, they instinctively convert documents to plain text, Markdown, or CSVs, and often rely heavily on brittle OCR engines or traditional computer-vision scripts despite possessing far stronger native visual capabilities (Anthropic, 2026; Bai et al., 2025; Google DeepMind, 2025, 2026; Meta AI, 2025, 2026; OpenAI, 2025, 2026; Zhu et al., 2025)—and even when looking at images directly, models remain prone to geometric blindness and strong text-prior bias (Rahmanzadehgervi et al., 2024; Vo et al., 2025).

Existing agentic benchmarks fail to expose this blind spot (Table 1; Section 5). Multi-file workplace evaluations such as GDPval (Patwardhan et al., 2025) and APEX-Agents (Vidgen et al., 2026), either allow task-critical signal to be extracted via scripts with negligible score loss, or focus on text- and spreadsheet-heavy domains. Meanwhile, visual QA suites (Chen et al., 2024b; Fu et al., 2024; Garre et al., 2026; Wang et al., 2024c; Yue et al., 2024b) test single-turn text responses without tool use,

[*]Equal contribution (co-first authors): Anima Singh (research & evaluation lead), Suhas Yogin (end-to-end dataset & release lead), and Weicheng Kuo (early data & task design lead). Core contributors are shown in **bold**, followed by contributors in alphabetical order and leadership.

*Corresponding author(s): {animasingh, suhasyogin, weicheng, rohandoshi, bencaine}@google.com*
© 2026 Google. All rights reserved

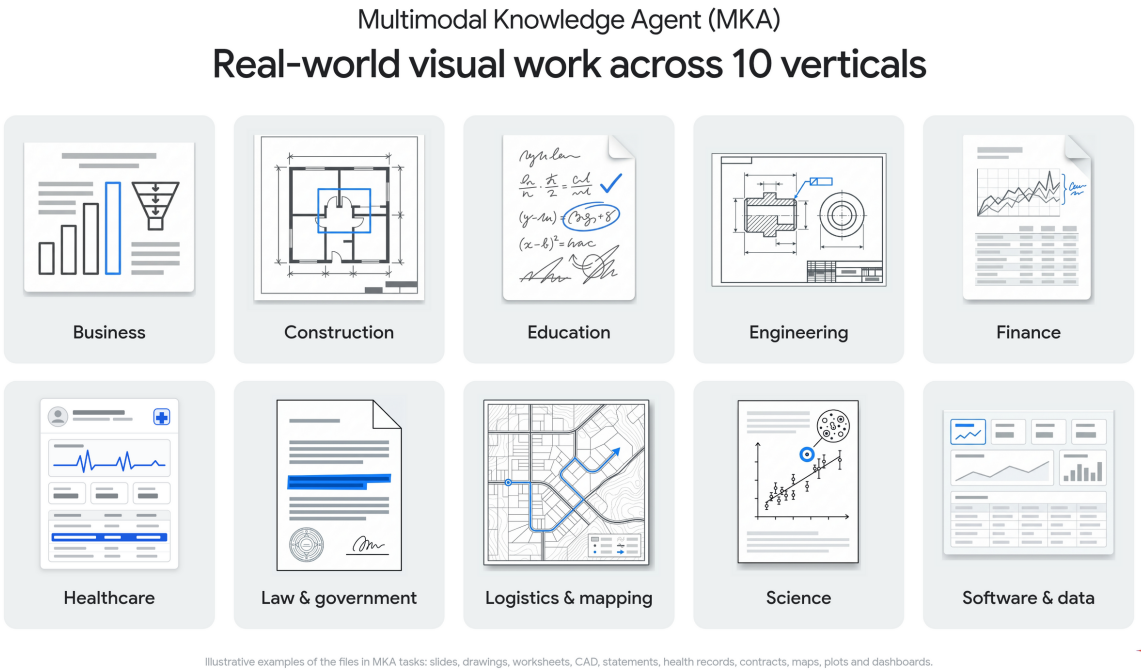



*Figure 1 | Overview of the 10 economic sectors in MKA (200 tasks total, 20 per sector), spanning blueprints, schematics, financial charts, clinical records, contracts, geospatial maps, scientific plots, and interactive dashboards.*

spreadsheet benchmarks (Ma et al., 2024b; Surge AI, 2026; Zhu et al., 2026) remain strictly tabular, browsing benchmarks (Li et al., 2025b; Wei et al., 2025) output short exact-match strings, and GUI benchmarks (Li et al., 2025a; Xie et al., 2024) entangle reasoning with pixel-level clicking.

To close this gap, we introduce **MKA (Multimodal Knowledge Agent)**, an end-to-end benchmark of 200 professional workflows across 10 evenly balanced sectors and 83 occupations (Figure 1; Section 2), alongside 10 disjoint open showcase tasks (Section 2.1). Each task provides 2 to 45 heterogeneous files (mean 7.3) with visual artifacts—such as scaled floor plans, electronics schematics, route maps, or radiographs—where flattening files to text destroys crucial information. Standalone file deliverables (`.xlsx`, `.pptx`, `.pdf`, `.docx`, `.html`, `.png`, `.txt`) are required in 88% of tasks, making vision essential both to read inputs and to create final outputs.

Controlled tool ablations in our standardized Stirrup harness (Artificial Analysis, 2025) show that vision in MKA is load-bearing. Removing image-viewing tools (`view_image`) from Gemini 4 Carbon leaves GDPval quality and compute unchanged, whereas on MKA it reduces Climb-WSR by ~5.9 percentage points and inflates agent turns by 1.28× and output tokens by 1.40× as the model falls back on coding workarounds. These workarounds recover part of the score on clean drawings but fail outright on photographs and medical images (Section 4.3).

To grade binary deliverables at scale without repeating the text-extraction mistake, we pair Stirrup with a sandboxed Agentic Autorater (Chen et al., 2024a; Zhuge et al., 2025) that visually renders candidate files and programmatically inspects formulas and document internals, reaching 96.2% pairwise criterion agreement with vetted human experts in a 50-task study ($\kappa$ = 0.91; task-level $r$ = 0.966; Section 3.3). Evaluating leading frontier models (Gemini 4 Carbon, Claude Fable 5.2, Claude Opus 5.5, and GPT-6 Astra) under identical harness budgets, Claude Fable 5.2 (56.6% Climb-WSR, 73.3% Overall WSR) and Gemini 4 Carbon (54.3% Climb-WSR, 71.9% Overall WSR) achieve

*Table 1 | Comparison of MKA against representative frontier benchmarks across multimodal, browsing, GUI, tabular, and professional knowledge-work evaluations.*

| Benchmark | Load-Bearing Vision | Multi-Step Tool Use | Heterogeneous Multi-File | Open-Ended Deliverables | Visual + Code Grading |
|---|---|---|---|---|---|
| MMMU-Pro / CharXiv (Wang et al., 2024c; Yue et al., 2024b) | Yes | No | No | No | No |
| GDP.pdf (Garre et al., 2026) | Yes | No | No (PDFs Only) | No | No |
| GDP.xlsx (Surge AI, 2026) | No | Yes | No (Single File) | No | No |
| BrowseComp / MM-BrowseComp (Li et al., 2025b; Wei et al., 2025) | Partial | Yes | No | No | No |
| OSWorld (Xie et al., 2024) | Yes (GUI) | Yes | Partial | Partial | No |
| SpreadsheetBench 2 (Zhu et al., 2026) | No (Inputs) | Yes | No (Tabular) | Yes (`.xlsx`) | Partial |
| APEX-Agents (Vidgen et al., 2026) | Partial | Yes | Yes | Partial | No |
| GDPval (Patwardhan et al., 2025) | No (Bypassable) | Yes | Yes | Yes | No |
| **MKA (Ours)** | **Yes** | **Yes** | **Yes** | **Yes (88%)** | **Yes** |

the strongest deliverable quality within statistical noise, with Carbon using 5.7× fewer tokens and 6.3× lower API cost (Section 4), while top Climb-WSR near 57% highlights substantial remaining headroom for genuinely multimodal agents.

## 2. The MKA Dataset

MKA consists of 200 real-world tasks written and verified by vetted domain professionals. The tasks are split evenly across 10 sectors (20 tasks each) and cover 83 occupations, including financial analysts, licensed architects, electrical and mechanical engineers, GIS surveyors, data scientists, attorneys and physicians. Each author has at least five years of professional experience, and every task was reviewed for realism and accuracy by a second expert with the same job profile.

Each task has a single prompt that specifies the user scenario, the requirements and the target deliverable. An expert spent 3 to 8 hours on each task: assembling a realistic multi-file input bundle, solving the task to produce golden reference deliverables, and writing a grading rubric with a golden justification for every criterion.

**Input files.** Each task includes 2 to 45 input files (1,459 in total; mean 7.30, median 6.0), and solving it requires combining information across visual, tabular, document and geospatial formats. Figure 2 breaks down the 1,459 input files by format category (percentages are shares of total input files):

- **PDF documents (43.0%, 628 files):** architectural elevations and floor plans, CAD and piping schematics, SEC Form 10-K filings, commercial leases, clinical operative reports and regulatory manuals (`.pdf`).
- **Images and scans (22.9%, 334 files):** floor plans, circuit and mechanical drawings, CT scans and radiographs, GIS route maps, digital elevation models and scanned handwritten forms (217 `.png`, 105 `.jpg`/`.jpeg`, 6 `.tif`, 3 `.gif`, 3 `.svg`).
- **Spreadsheets (10.9%, 159 files):** multi-tab Excel workbooks containing balance sheets and operating models, claims and exposure registers, engineering bills of materials and project

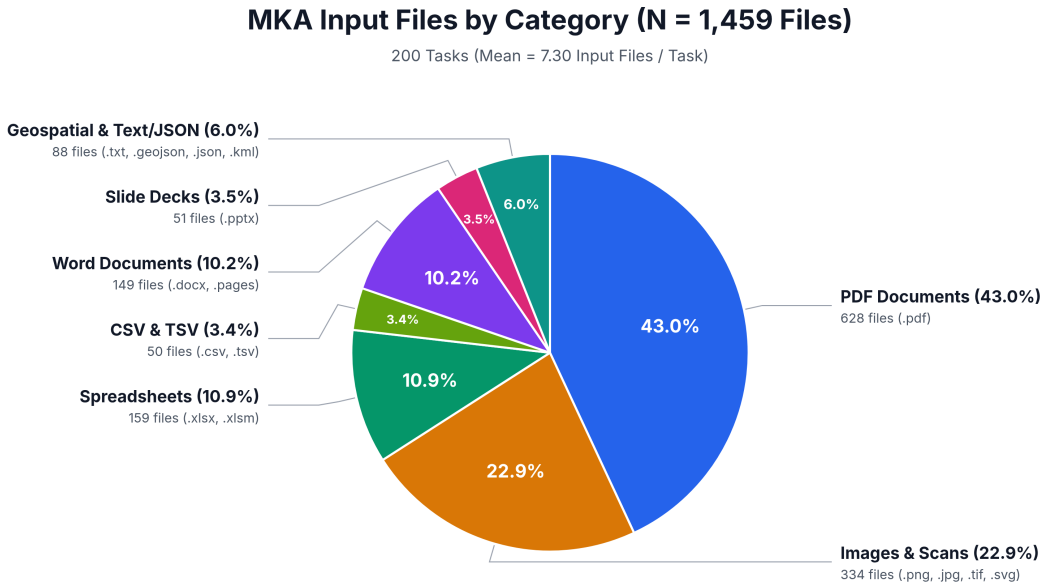



*Figure 2 | Distribution of the 1,459 input files across the 200 MKA tasks by format category.*

Gantt charts (158 `.xlsx`, 1 `.xlsm`).

- **Word documents (10.2%, 149 files):** standard operating procedures, purchase agreements, clinical coding crosswalks, witness statements and report templates (148 `.docx`, 1 `.pages`).
- **Geospatial and text/JSON (6.0%, 88 files):** municipal boundary and route layers, LiDAR and cartography specifications, EEG metadata, email threads and correspondence logs (42 `.txt`, 21 `.geojson`, 10 `.json`, 3 `.md`, 3 `.eml`, 2 `.html`, 2 `.kml` and 5 shapefile/world-file/script assets).
- **Slide decks (3.5%, 51 files):** executive board briefings, investment mandates, network architecture boards and presentation templates (`.pptx`).
- **CSV and TSV (3.4%, 50 files):** delimited telemetry logs, treasury rate series, fleet dispatch manifests and clinical trial tables (47 `.csv`, 3 `.tsv`).

**Deliverables.** Of the 200 tasks, 176 (88.0%) require the agent to create or modify a standalone file, while the remaining 24 (12.0%) are answered directly in the text response. Figure 3 shows the distribution of primary deliverables across all 200 tasks (percentages sum to 100.0%):

- **PDF reports (22.5%, 45 tasks):** formatted engineering evaluations, property appraisals, clinical SOPs and audit memoranda (`.pdf`).
- **Word documents (21.0%, 42 tasks):** structured technical reports, legal and regulatory reviews, and feasibility studies (`.docx`).
- **Presentations (17.0%, 34 tasks):** multi-slide executive briefings, investment proposals and training decks (`.pptx`).
- **Spreadsheets (16.0%, 32 tasks):** multi-tab financial projections, tax and accounting schedules, and engineering workbooks (`.xlsx`, `.xlsm`).
- **Text reports (15.0%, 30 tasks):** structured written analyses returned either directly in the text response (24 tasks, 12.0%) or as standalone plain-text files (6 tasks, 3.0%, `.txt`).
- **Images and charts (4.5%, 9 tasks):** high-resolution maps, diagrams and visual figures (7 `.png`, 2 `.jpg`).

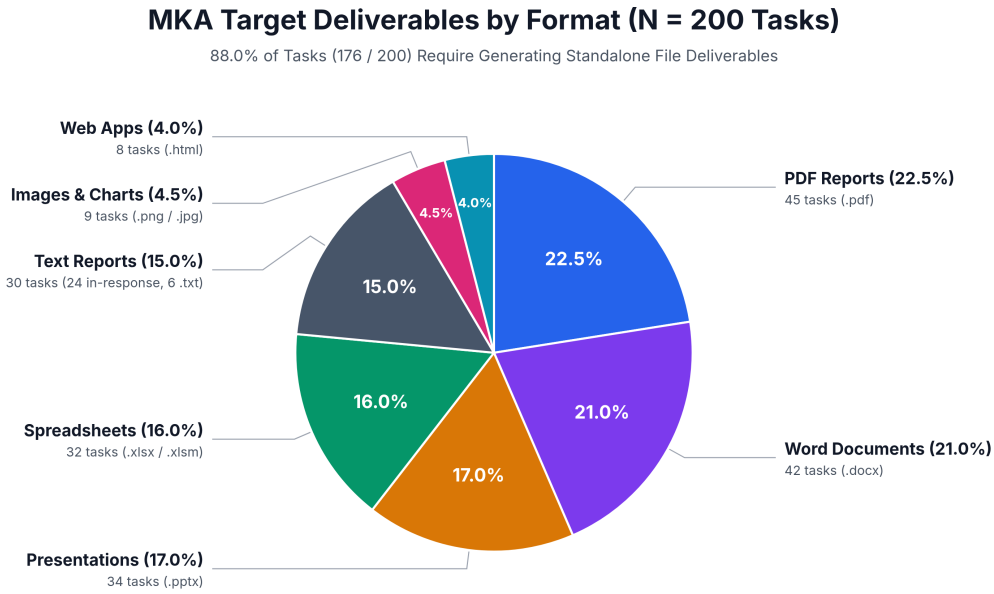



*Figure 3 | Distribution of primary deliverable formats across the 200 MKA tasks.*

- **Web apps (4.0%, 8 tasks):** self-contained interactive HTML dashboards and visual applications (`.html`).

**Rubrics.** Across the 200 tasks, MKA contains 14,284 rubric criteria (mean 71.4 per task, median 68.5, range 20–150):

- **Target:** 91.3% of criteria (13,037) evaluate the 176 tasks requiring standalone file deliverables, while 8.7% (1,247) evaluate the 24 text-response tasks (1,735 criteria, or 12.1%, when including the 6 standalone `.txt` tasks).
- **Weights and priority:** each criterion $j$ of task $i$ has an integer weight $w_{ij} \in \{1, \ldots, 5\}$ (21.6%, 26.2%, 26.1%, 21.4% and 4.7% for weights 1–5, respectively) used to compute the Weighted Satisfaction Rate (Section 3.4), and is labeled as a primary objective (94.5%, 13,497 criteria) or a secondary objective (5.5%, 787 criteria).
- **Dependencies:** 29.5% of criteria (4,217 across 190 tasks) list explicit prerequisite criteria in their metadata, forming a directed acyclic graph (DAG) within each task that records how downstream conclusions depend on upstream visual extractions and calculations (during grading, the autorater evaluates each criterion independently and WSR aggregates all criteria by $w_{ij}$).
- **Golden justifications:** 99.6% of criteria (14,229 of 14,284; 100% across 199 of the 200 tasks) include an expert-written golden justification that cites the source location of the ground truth and, where applicable, its derivation.

**Sectors.** Figure 1 illustrates how the 200 tasks are distributed evenly across 10 sectors (20 tasks, or 10.0%, per sector) and 83 occupations. The bullets below highlight a few reference job profiles in each sector:

- **Business:** management consultants, marketing and media analysts, customer support specialists, revenue management analysts and procurement specialists.

- **Construction:** licensed architects, construction estimators, quantity surveyors, real estate appraisers and civil or fire protection engineers.
- **Education:** K–12 and university instructors, curriculum compliance analysts, academic moderators, school psychologists and training specialists.
- **Engineering:** mechanical and CAD engineers, electrical and RF engineers, chemical process engineers, power systems engineers and manufacturing quality engineers.
- **Finance:** financial and investment analysts, accountants and auditors, property and marine underwriters, credit analysts and tax preparers.
- **Healthcare:** diagnostic and pediatric radiologists, radiation oncologists, telestroke neurologists, hospital administrators and medical coding or claims specialists.
- **Law and government:** immigration, corporate, real estate and patent attorneys, regulatory analysts, municipal procurement officers and government administrators.
- **Logistics and mapping:** GIS analysts, land surveyors, urban and regional transport planners, precision agriculture analysts and logistics analysts.
- **Science:** clinical trial researchers, medicinal and analytical chemists, molecular and wildlife biologists, materials scientists and EEG analysts.
- **Software and data:** front-end engineers, data analysts, data engineers, software QA engineers and radio or network engineers.

**Reference examples.** Figure 4 shows three example tasks from the open MKA-Showcase split (Section 2.1):

- **Workforce planning analyst (Business).** The agent conducts a fiscal-year department staffing review across 6 input files: quarterly voluntary and involuntary leave alluvial charts (`.png`), baseline headcount records (`.csv`), staffing and tie-break rules (`.pdf`), a corporate branding template (`.xlsx`) and a company logo (`.jpg`). It must trace leave flows between 15 departments across both charts and produce a branded 2-sheet workbook (`.xlsx`) ranking each department's least and most frequent feeders (90 rubric criteria).
- **GIS analyst (Logistics and mapping).** The agent prepares a municipal rooftop solar assessment across 9 geospatial and tabular files. It identifies the study area (City of Nottingham) from an unlabeled reference map (`.jpeg`), joins administrative, building-footprint and heritage layers (`.geojson`) against 1 m LiDAR elevation and solar-irradiance rasters (`.tif`), a council asset register (`.xlsx`) and screening rules (`.txt`), and produces an interleaved report (`.docx`) with maps and a ranked table of the top five council buildings (62 rubric criteria).
- **Mechanical engineering instructor (Engineering).** The agent builds a nonlinear dynamics lab deck across 8 input files. It reads bilateral backlash geometry and parameters from a mechanical schematic (`.pdf`), follows a subspace-switching flowchart and multiple-shooting reference paper (`.pdf`), applies presentation specifications (`.docx`, `.pdf`) and a slide template (`.pptx`), and solves the periodic boundary-value problem to produce a 6-slide presentation (`.pptx`) with phase-portrait charts (54 rubric criteria).

### 2.1. Public Leaderboard

To support independent evaluation while preventing benchmark contamination, MKA is accompanied by two disjoint task sets. **MKA (200 tasks)** is the private evaluation suite reported in this paper, hosted and graded independently on a public leaderboard by Artificial Analysis using the open-source Stirrup framework (Section 3.1). **MKA-Showcase (10 tasks)** is a separate, publicly downloadable release of one task per sector with full prompts, input files, and grading rubrics.

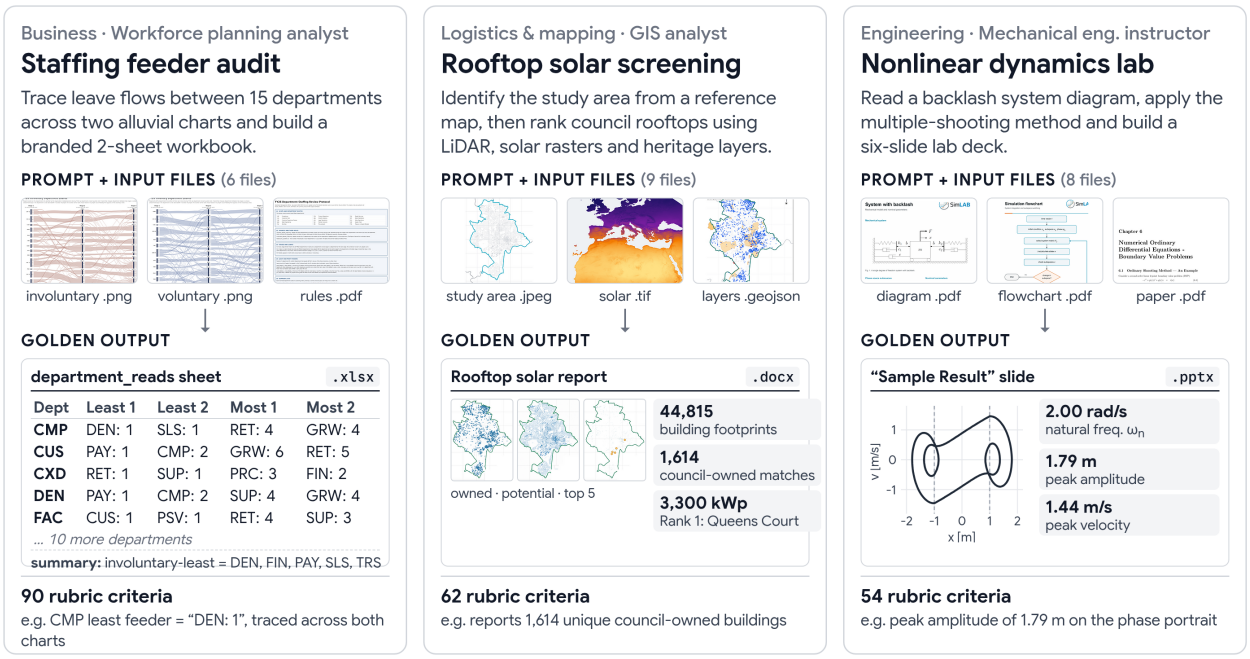



*Figure 4 | Three example tasks from the open MKA-Showcase split. Each card shows the sector and expert role, a summary of the prompt, three of the task's input files, a preview of the human golden deliverable, and the number of rubric criteria with one example criterion: a department staffing feeder audit (workforce planning analyst), a council rooftop solar screening (GIS analyst), and a nonlinear dynamics lab deck (mechanical engineering instructor).*

## 3. Evaluation Setup

### 3.1. Agentic Harness

We evaluate every model in Stirrup (`v0.2.1`) (Artificial Analysis, 2025), an open-source, lightweight agent framework maintained by Artificial Analysis. The harness gives the agent the task prompt and the paths of the input files as text, so the agent must open and inspect each file and build the final deliverables through tool calls. All models use the same harness, system instructions, tool set, container image and resource budgets; only the model endpoint changes. An externally maintained, minimal harness avoids model-specific scaffolding, so deliverable quality and token efficiency are directly comparable across model families.

**Agent loop.** Stirrup runs without human intervention until the agent calls `finish` or reaches the limit of 250 turns, with up to a 1M-token context window if available, and an 8-hour wall-clock limit per task. At 70% of the model's context length, compaction is triggered. If the model replies without a tool call, the harness prompts it to continue. In the final 20 turns, the harness appends a warning with the number of remaining turns so the agent can finish and submit its deliverables.

**Sandbox.** Each task runs in a fresh, isolated Linux container (4 vCPUs; 128 MB base RAM dynamically bursting up to 16 GB) based on a Debian container with Python 3.13 (`python:3.13-slim-trixie`), with working directory `/workspace` and input files staged under `/workspace/input`. The sandbox has internet access. The image includes common document, PDF, OCR and rendering tools, including LibreOffice, Tesseract, `pymupdf`, `weasyprint` and Python Office libraries, so agents

can parse inputs and build deliverables without installing software.

**Tools.** Stirrup exposes four tools: `code_exec`, `view_image`, `web_search` and `finish`. `code_exec` runs bash commands and Python scripts (300 s per-command timeout), and `web_search` issues web queries (180 s timeout). `view_image` loads a raster image (`.png`, `.jpg`, `.webp`, `.gif`, `.bmp` or `.tiff`) into the model's visual context. To inspect a document or slide deck visually, the agent renders pages to `.png` with `code_exec` using command-line tools such as LibreOffice and `pdftoppm`, and views the pages it needs with `view_image`. `view_image` is designed to respect the native image resolution of the output file. At the end of a trajectory, `finish` checks that every declared deliverable exists and ends the session.

### 3.2. Agentic Autorater

Most MKA tasks (88.0%; 91.3% of rubric criteria) require standalone files such as spreadsheets (`.xlsx`), slide decks (`.pptx`), documents (`.docx`), PDFs or images. MKA rubrics check both visual layout and underlying file internals such as spreadsheet formulas. Static text extraction omits charts and formatting, while static page renders hide cell formulas and can clip wide sheets. To give the grader direct access to both, we use an agentic, multimodal judge that can inspect rendered pages and run code to query file contents.

**Grading harness.** The Agentic Autorater is also a Stirrup agent. It runs in its own fresh container with `code_exec`, `view_image` and a structured `submit_verdicts` finish tool, and without web access. Its `/workspace` contains only the candidate's deliverable files and final text response. The autorater does not receive the task's input files: MKA rubrics are self-contained, with each criterion stating the expected value, tolerance or property directly, so grading checks the deliverable against the rubric without re-solving the task. In a single session, the autorater checks these outputs against the verbatim task prompt and rubric and returns, for each criterion, a pass/fail verdict, a justification and the files it cited (prompt in Appendix A). We use OpenAI GPT-6 Luna as the autorater model. Using a judge from outside the Google model family avoids favoring Gemini; we also see no sign of same-family preference, as GPT-6 Astra ranks last among the evaluated models.

### 3.3. Agreement Between the Autorater and Human Raters

We compare the Agentic Autorater's verdicts with independent human ratings using the pairwise agreement metric of GDPval (Patwardhan et al., 2025, App. A.6.2). For binary pass/fail verdicts, this metric measures the fraction of rater pairs that assign the same verdict on a criterion, averaged across criteria, and compares human–autorater agreement against the inter-human agreement ceiling (Eq. 1). Matching Eq. (2), we weight criteria by $w_{ij}$ within each task ($\sum_j w_{ij} a_{ij} / \sum_j w_{ij}$) and average equally across the tasks to obtain $A^{\mathrm{HA}}$ and the inter-human ceiling $A^{\mathrm{HH}}$, with 95% CIs from resampling the tasks with replacement.

**Human study.** Across 50 MKA tasks spanning 9 verticals, we pair each task with one deliverable produced by the frontier models. Three domain experts, blind to model identity, grade each deliverable independently: for every rubric criterion, they inspect the task inputs and the deliverable and record a pass/fail verdict with a written rationale. Unlike the autorater, human raters also see the task inputs, so agreement between the two also tests whether the rubrics are sufficient for grading on their own. To focus expert annotation on substantive factual, quantitative, and structural requirements, we filter

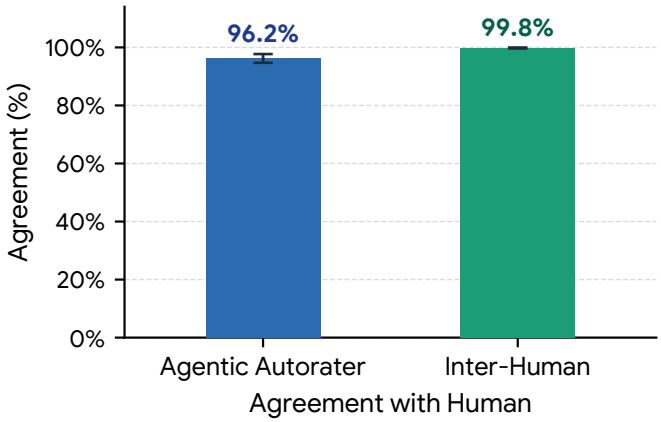



*Figure 5 | Pairwise criterion agreement with human domain raters (error bars: 95% task-level bootstrap CIs). The Agentic Autorater reaches 96.2% agreement ($A^{\mathrm{HA}}$), matching 96.4% of the 99.8% inter-human ceiling ($A^{\mathrm{HH}}$).*

to high-weight rubric criteria (weights 3–5) and trim redundant passing checks on high-scoring tasks to preserve a balanced pass/fail mix. We also run the autorater three times per deliverable.

**Agreement metrics.** For each criterion $c$, we compute human–autorater agreement as

$$A_c^{\mathrm{HA}} = \mathbb{E}_{i,j}\left[1 - |H_{c,i} - R_{c,j}|\right], \tag{1}$$

where $H_{c,i}, R_{c,j} \in \{0, 1\}$ are the verdicts of human rater $i$ and autorater run $j$ on criterion $c$. Weighting $A_c^{\mathrm{HA}}$ by $w_{ij}$ within each task and averaging equally across tasks yields $A^{\mathrm{HA}}$. Analogously, we compute the human ceiling $A^{\mathrm{HH}}$ over distinct pairs of human raters ($i \neq i'$), with 95% confidence intervals from a bootstrap over tasks.

**Results.** The Agentic Autorater reaches $A^{\mathrm{HA}} = 96.2\%$ $[94.7, 97.6]$ alongside an inter-human agreement ceiling of $A^{\mathrm{HH}} = 99.8\%$ $[99.6, 100.0]$—matching 96.4% of human–human agreement (Figure 5; 99.7% of criteria are unanimous across all three raters). Accounting for the 69.1%/69.9% weighted pass rates (expected chance agreement 57.6%), weighted pairwise Cohen's $\kappa$ is 0.911 $[0.873, 0.943]$ (0.919 $[0.882, 0.950]$ on majority verdicts; inter-human Fleiss' $\kappa = 0.996$). Such near-unanimous expert agreement is expected because MKA rubrics specify objective, factual checks (computed values, tolerances, required fields, and explicit formatting rules). At the task level, autorater weighted rubric scores track human scores closely, with Pearson $r = 0.966$ and Spearman $\rho = 0.931$.

**Error breakdown.** Treating the human majority verdict as ground truth yields an autorater error rate of 3.8% $[2.4, 5.4]$ per criterion (about one error in 26 criteria). Table 2 decomposes this 3.8% error rate into *false passes* (the deliverable is wrong or missing a required item, yet the autorater marks the criterion passed; 2.4%) and *false fails* (the deliverable is correct, yet the autorater marks the criterion failed; 1.4%). False passes mainly reflect boundary leniency, such as crediting alternative or non-primary values when the model's committed answer differs, or accepting semantically equivalent content when a required label is omitted. Conversely, false fails stem from overly strict string or structural matching where domain experts accept equivalent formulations, or from static code inspection of interactive deliverables without live execution.

*Table 2 | Decomposition of the 3.8% per-criterion autorater error rate against the human majority verdict (95% task-level bootstrap CIs). Conditional rates normalize by the human-failed (34.2%) and human-passed (65.8%) base rates.*

| Error Type | Definition (vs. Human Majority) | % of All Criteria | Conditional Rate (%) |
|---|---|---|---|
| False pass | Human fails, autorater passes | 2.4 [1.3, 3.8] | 7.1 [3.6, 11.4] |
| False fail | Human passes, autorater fails | 1.4 [0.8, 2.0] | 2.1 [1.2, 3.1] |
| **Total error** | Any disagreement with human majority | **3.8** [2.4, 5.4] | — |

### 3.4. Quality and Efficiency Metrics

We evaluate each model on deliverable quality and token efficiency across all 200 tasks. All metrics are task-level means with tasks weighted equally. Rollouts that fail due to transient infrastructure or API errors are re-run. Sessions that hit the turn or wall-clock limit are not re-run and are graded on any outputs produced before termination; rollouts that produce no output (such as safety refusals) score zero on all criteria.

**Weighted Satisfaction Rate (WSR).** Each task $i$ has rubric criteria $j$ with expert weights $w_{ij} \in \{1, \ldots, 5\}$ and binary autorater verdicts $s_{ij} \in \{0, 1\}$. Task WSR is the weighted share of satisfied criteria:

$$\mathrm{WSR}_i = \frac{\sum_j w_{ij}\, s_{ij}}{\sum_j w_{ij}}. \tag{2}$$

WSR scores each criterion directly from $w_{ij}$ and $s_{ij}$ without gating on the primary/secondary labels or prerequisite DAGs from Section 2; those annotations structure rubric authoring and quality control, and we retain them in the released metadata for diagnostic analysis and optional gated scoring. We do not report task-level all-pass rate because it is brittle and has high variance: with 70+ criteria per task, a single missed minor check or autorater flip zeroes out downstream criteria or the entire task.

**Climb and regression splits.** Domain experts author rubrics covering every task requirement, including basic formatting checks that current models pass routinely. Using a fixed panel of four prior-generation models (Gemini 3.8 Flash, Gemini 3.6 Flash, Claude Sonnet 5.5 and GPT-6.1 Sol), we label a criterion *regression* if every model in the panel passes it in at least 2 of 3 runs (6,396 criteria across 180 tasks), and *climb* otherwise (7,888 criteria across all 200 tasks). Tasks that consistently run out of context for the model panel are labeled all-*climb*. Because this split is derived from autorater verdicts, a small fraction of criteria the panel satisfies may also be labeled *climb* due to judge false fails (Section 3.3); the 2-of-3 rule dampens single-run flips, and the frozen split applies equally to all evaluated models. Computing Eq. (2) over each subset yields three task-level scores: **Climb-WSR** (headline; frontier capability), **Regression-WSR** (retention of skills that prior-generation models already have, averaged over the 180 tasks with regression criteria) and **Overall WSR** (overall deliverable quality).

## 4. Experiments and Results

We evaluate Gemini 4 Carbon alongside leading external frontier models: Claude Fable 5.2, Claude Opus 5.5 and GPT-6 Astra. Each model runs once per task in the same Stirrup harness, container environment and turn budget (Section 3.1) and is graded in a single pass by the Agentic

Autorater (Section 3.2). None of the evaluated models is part of the panel used to define the climb and regression splits (Section 3.4).

### 4.1. Model Quality Comparison

Table 3 and Figure 6a compare task-level Climb-WSR, Overall WSR and Regression-WSR. All four models saturate Regression-WSR above 95% (95.17%–96.36%), confirming that frontier differences are concentrated in Climb-WSR. Claude Fable 5.2 (56.56% Climb-WSR, 73.26% Overall WSR) and Gemini 4 Carbon (54.34% Climb-WSR, 71.89% Overall WSR) achieve the highest deliverable quality. Under paired task-level bootstrap, the top three models—Claude Fable 5.2, Gemini 4 Carbon, and Claude Opus 5.5 (53.01% Climb-WSR, 70.98% Overall WSR)—are statistically indistinguishable on Climb-WSR (Fable 5.2 vs. Carbon: +2.22 pp, 95% CI $[-0.94, +5.49]$; Carbon vs. Opus 5.5: +1.33 pp, 95% CI $[-1.70, +4.41]$), whereas both Fable 5.2 and Gemini 4 Carbon significantly outperform GPT-6 Astra (50.24% Climb-WSR, 69.34% Overall WSR; Carbon vs. Astra: +4.10 pp Climb-WSR, 95% CI $[+1.00, +7.11]$). Even for the strongest frontier models, Climb-WSR remains around 54%–57%, leaving substantial headroom to evaluate multimodal agent progress.

*Table 3 | Deliverable quality on MKA (task-level means ± 95% CI, %).*

| Model | Climb-WSR | Overall WSR | Regression-WSR |
|---|---|---|---|
| Claude Fable 5.2 | **56.56** ±3.63 | **73.26** ±2.45 | 95.71 ±1.68 |
| Gemini 4 Carbon | 54.34 ±3.24 | 71.89 ±2.15 | **96.36** ±0.91 |
| Claude Opus 5.5 | 53.01 ±3.54 | 70.98 ±2.46 | 95.81 ±1.32 |
| GPT-6 Astra | 50.24 ±3.25 | 69.34 ±2.10 | 95.17 ±1.07 |

*Table 4 | Mean per-task token usage, agent turns, runtime and list-price API cost (with prompt caching) on MKA. Output tokens sum reasoning and answer tokens for every model.*

| Model | Total Tok. | Input Tok. | Output Tok. | Turns | Time (min) | API Cost |
|---|---|---|---|---|---|---|
| Claude Fable 5.2 | 40.48M | 40.10M | 378.9k | 128.8 | 83.2 | $35.28 |
| Gemini 4 Carbon | 7.05M | 6.92M | 129.5k | 46.6 | 29.2 | **$5.63** |
| Claude Opus 5.5 | 19.96M | 19.75M | 208.8k | 91.4 | 33.8 | $10.29 |
| GPT-6 Astra | **2.43M** | **2.39M** | **42.0k** | **29.0** | **18.5** | $6.03 |

### 4.2. Model Efficiency Comparison

Table 4 and Figures 6b–d compare mean per-task token usage, trajectory length, runtime and estimated API cost at list prices (accounting for each provider's prompt-cache read/write pricing, where 93.7%–97.7% of cumulative input tokens are served from cache; output tokens include both reasoning and visible answer tokens for all models). Trajectory efficiency varies by over 16× across frontier models:

- **Claude Fable 5.2** consumes 5.74× as many total tokens as Gemini 4 Carbon (40.48M vs. 7.05M), 2.93× as many output tokens (378.9k vs. 129.5k, with 286.8k vs. 76.6k reasoning tokens), 2.76× as many turns (128.8 vs. 46.6), and 2.85× the wall-clock time (83.2 vs. 29.2 minutes per task). At provider list prices, this corresponds to a 6.26× higher API cost ($35.28 vs. $5.63 per task) for a statistically non-significant +2.22 pp Climb-WSR difference.
- **Claude Opus 5.5** likewise uses 2.83× as many total tokens (19.96M), 1.61× as many output tokens (208.8k over 91.4 turns) and 1.83× higher API cost ($10.29 per task) than Gemini 4 Carbon while scoring 1.33 pp lower on Climb-WSR.

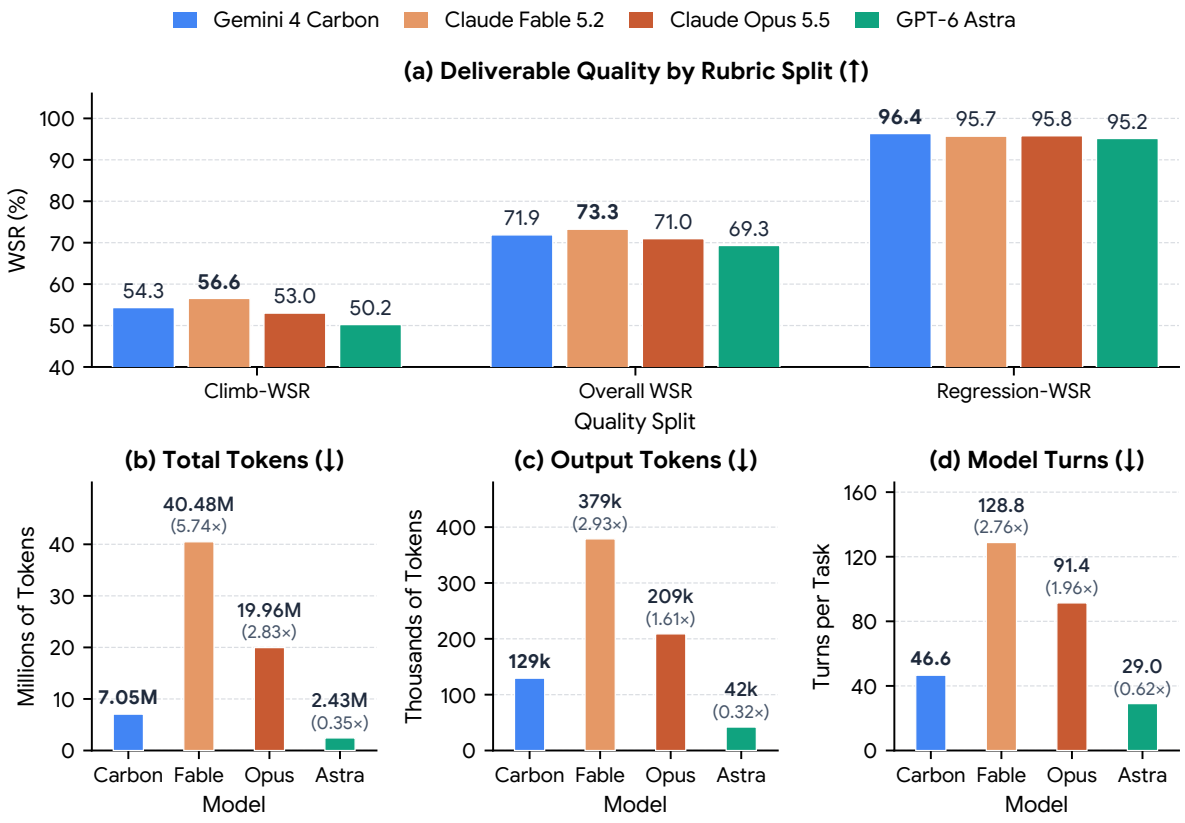



*Figure 6 | Deliverable quality across rubric splits (**a**; task-level means) and mean total tokens, output tokens and model turns per task (**b–d**; multipliers relative to Gemini 4 Carbon).*

- **GPT-6 Astra** uses the fewest tokens and turns (0.35× Carbon's total tokens: 2.43M total and 42.0k output over 29.0 turns in 18.5 minutes), yet because of higher per-token list pricing it still costs slightly more per task than Gemini 4 Carbon ($6.03 vs. $5.63) while scoring 4.10 pp lower on Climb-WSR.

Figure 7 plots Climb-WSR against total token usage: GPT-6 Astra, Gemini 4 Carbon and Claude Fable 5.2 form the token Pareto frontier, with Gemini 4 Carbon offering the best quality-per-token among the top scoring models, and achieving the lowest API cost per task ($5.63).

### 4.3. Role of Visual Perception

To test how much MKA relies on genuine visual perception rather than text extraction, we evaluate Gemini 4 Carbon with and without the `view_image` tool under the identical 250-turn Stirrup setup, and run the same ablation on GDPval (Patwardhan et al., 2025). Without `view_image`, the agent retains `code_exec`, `web_search`, and all pre-installed parsing, OCR, and computer-vision libraries, so it can still extract embedded document text, parse spreadsheets, and write custom image-analysis scripts. On GDPval, we grade both arms with the same Agentic Autorater (Section 3.2) against the expert-authored rubrics released with the public GDPval gold subset and report Overall WSR (88.1% w/ vs. 88.2% w/o `view_image`).

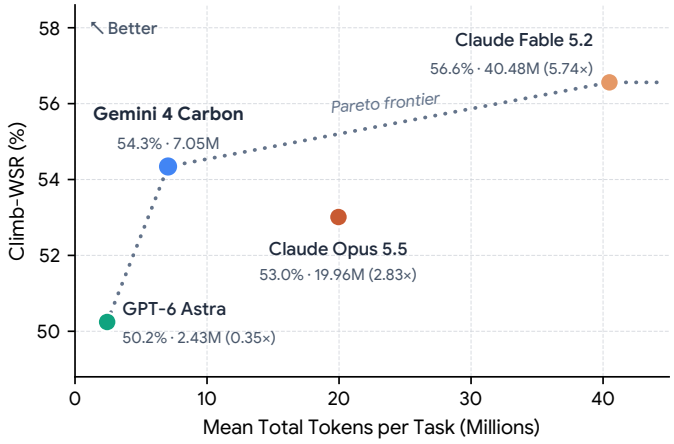



*Figure 7 | Deliverable quality (Climb-WSR) versus mean total tokens per task across MKA tasks (upper-left is better). GPT-6 Astra, Gemini 4 Carbon and Claude Fable 5.2 trace the token Pareto frontier, with Gemini 4 Carbon using fewer tokens than Claude Opus 5.5 at similar quality, and sitting at the knee before Fable 5.2's 5.74× token jump.*

*Table 5 | Tool ablation (w/ vs. w/o `view_image`) for Gemini 4 Carbon on MKA and GDPval. Removing `view_image` leaves GDPval unchanged in both quality and cost, whereas MKA suffers a significant Climb-WSR drop alongside steep token and turn inflation.*

| | **Quality Δ (w/o − w/ `view_image`, pp)** | | | **Compute Ratio (w/o / w/ `view_image`)** | | |
|---|---|---|---|---|---|---|
| **Benchmark** | **Climb-WSR** | **Overall WSR** | **Reg.-WSR** | **Input Tok.** | **Output Tok.** | **Turns** |
| MKA | **−5.94** | **−4.58** | **−1.96** | **1.85×** | **1.40×** | **1.28×** |
| GDPval | — | +0.08 (ns) | — | 0.89× | 1.05× | 0.95× |

**MKA vs. GDPval.** Table 5 shows a sharp split between the two benchmarks. On GDPval, removing `view_image` changes neither the deliverable's analytical quality (+0.08 pp Overall WSR, 95% CI [−0.62, +0.78]) nor compute (0.95× turns, 1.05× output tokens, and 0.89× input tokens), because task-critical inputs reside in spreadsheets and machine-readable text. On MKA, removing `view_image` reduces Overall WSR by 4.58 pp ([−5.51, −3.65]) and Climb-WSR by 5.94 pp ([−7.39, −4.49])—more than 3× the drop on Regression-WSR (−1.96 pp)—while inflating trajectories to 1.28× model turns, 1.40× output tokens, and 1.85× input tokens.

**Why strong coding can compensate—but cannot replace—vision.** Although a −5.94 pp Climb-WSR drop is substantial and statistically significant, it is smaller than one might expect on a visually grounded benchmark. Inspecting trajectories without `view_image` reveals that Gemini 4 Carbon partially compensates through pure persistence using code to create workarounds. Surprisingly, it goes far beyond standard text extraction and OCR, and starts rendering image crops as ASCII art or unicode Braille blocks in its terminal, or falls back to complicated and bespoke classical computer vision pipelines with OpenCV. Two case studies (Appendix B) illustrate the resulting split:

- **Partial recovery at steep cost:** On clean schematics (e.g., mechanical CAD drawings), ASCII rendering and OpenCV measurements recover primary geometry in some runs, but miss fine clearance hazards and require 1.8×–4.6× more tokens and turns.
- **Outright perceptual failure:** On natural photographs and medical imaging (e.g., road-

inspection photos, clinical ultrasounds and radiographs, or circuit-board photos), dozens of OpenCV and ASCII-art scripts still misread subtle visual structure, dropping Climb-WSR by 34–72 pp despite 1.8×–5.8× higher compute.

## 5. Related Work

MKA builds on a long tradition of benchmarks that have driven progress across two largely separate tracks: static multimodal perception over documents, charts, and diagrams, and autonomous tool use in coding, web, and office environments. Below, we situate MKA alongside prior work across workplace deliverables, spreadsheet workflows, multimodal question answering, active visual reasoning, agentic search, and computer-use grounding (summarized in Table 1).

**Workplace Deliverables.** Building on repository-level coding and CLI benchmarks (Jimenez et al., 2024; Merrill et al., 2026; Yang et al., 2025), multi-application office and simulated company environments (Wang et al., 2024b; Xu et al., 2024), and domain-expert rubric evaluations (Arora et al., 2025; Starace et al., 2025; Vidgen et al., 2025), recent suites evaluate agents on realistic project deliverables. APEX-Agents (Vidgen et al., 2026) places agents in multi-file worlds across banking, consulting, and law, grading outputs via per-criterion LLM-judge calls (`Pass@1`). GDPval (Patwardhan et al., 2025) spans 44 occupations and grades deliverables primarily via blinded pairwise preference. Although its workspaces include PDFs and slide decks, task-critical signal can largely be extracted via Python scripts and OCR without visual inspection.

**Tabular and Spreadsheet Workflows.** Dedicated spreadsheet benchmarks isolate multi-step code execution over complex workbooks without requiring visual perception or non-tabular deliverables. GDP.xlsx (Surge AI, 2026) equips agents with sandboxed Python tools over a single workbook to produce a text answer, while SpreadsheetBench 2 (Ma et al., 2024b; Zhu et al., 2026) evaluates `.xlsx` modeling, debugging, and chart creation without cross-modal inputs.

**Multimodal and Document Question Answering.** Foundational document and chart QA benchmarks (Ma et al., 2024a; Masry et al., 2022; Mathew et al., 2021) and harder frontier suites such as MMMU (Yue et al., 2024a), CharXiv (Wang et al., 2024c), and MMMU-Pro (Yue et al., 2024b) require strong visual perception, but present pre-selected images or pages in a single, tool-free turn and grade multiple-choice or short text answers. Diagnostic suites including MMStar (Chen et al., 2024b), BLINK (Fu et al., 2024), VLMs Are Blind (Rahmanzadehgervi et al., 2024), and VLMs Are Biased (Vo et al., 2025) further show that VLMs often bypass pixels via linguistic priors or fail at basic 2D geometric tracing. Extending document QA to enterprise files, Surge AI's GDP.pdf (Garre et al., 2026) evaluates atomic criterion rubrics over dense, multi-document professional PDFs, yet still pre-attaches source PDFs in a single-turn prompt and grades a text response via a text-only judge—omitting iterative tool use, cross-format synthesis, and binary deliverable verification.

**Frontier VLMs and Active Visual Tool Use.** Frontier multimodal models across proprietary (Anthropic, 2026; Google DeepMind, 2025; Meta AI, 2026; OpenAI, 2026) and open-weight (Bai et al., 2025; Meta AI, 2025; Zhu et al., 2025) families pair native visual perception with code-driven agent scaffolds (Wang et al., 2024a), and recent models across labs—from Google's Agentic Vision (Google DeepMind, 2026) to Claude (Anthropic, 2026), Meta Muse (Meta AI, 2026), Qwen (Bai et al., 2025), and OpenAI o3 (OpenAI, 2025)—actively inspect visual inputs by cropping, zooming, and transforming images inside chain-of-thought. MKA tests whether these active perceptual capabilities transfer

to multi-file enterprise workspaces where agents must choose when to write code versus when to inspect pixels.

**Agentic Search and Retrieval.** Following general tool-use assistants such as GAIA (Mialon et al., 2024), deep-research benchmarks evaluate multi-step web search over text (BrowseComp; Wei et al. 2025) and visual web content (MM-BrowseComp; Li et al. 2025b). By design, however, these benchmarks constrain outputs to a single short string or number to permit exact-match grading, leaving open-ended multi-file synthesis untested.

**GUI and Computer-Use Environments.** Browser and desktop benchmarks (Li et al., 2025a; Xie et al., 2024) evaluate agents acting through low-level pixel coordinates (`mouse_move`, `click(x, y)`) or DOM actions, verifying completion via deterministic post-hoc scripts that check exact file diffs, DOM states, or database records. While effective for closed-form UI tasks, model failures are dominated by brittle interface grounding, and deterministic state scripts cannot evaluate open-ended reports, executive slide decks, or valid layout trade-offs. Building on LLM-as-a-judge (Zheng et al., 2023), MLLM-as-a-Judge (Chen et al., 2024a), and Agent-as-a-Judge (Zhuge et al., 2025) paradigms alongside classical inter-rater agreement (Cohen, 1960; Fleiss, 1971), MKA decouples domain-level multimodal reasoning from GUI clicking while grading binary deliverables with a sandboxed visual and programmatic autorater.

## 6. Limitations

Each model runs once per task and is graded in a single autorater pass, so our confidence intervals capture task sampling but not run-to-run agent or grader variance. The human agreement study covers 50 tasks and focuses on high-weight criteria (weights 3–5) with a balanced pass/fail mix, so the 96.2% agreement may not transfer exactly to low-weight formatting checks. The vision ablation is run on Gemini 4 Carbon only; other models may rely on vision to different degrees. The sandbox has internet access, which mirrors real deployments but may let agents retrieve some information online instead of extracting it visually. The climb/regression panel includes two earlier Gemini models, so the split reflects what that panel already solves. Finally, API costs use provider list prices at the time of writing and will change.

## 7. Conclusion

Real-world knowledge work is inherently visual, requiring professionals to reason over blueprints, schematics, maps, charts, and scanned documents while verifying the layout of the files they produce. MKA brings this reality into agent evaluation across 200 multi-file workflows and 10 sectors, paired with an Agentic Autorater that matches human domain experts on 96.2% of criterion judgments in our expert study ($\kappa = 0.91$). Our findings show that native visual perception drives not only deliverable quality, but also compute efficiency: removing image-viewing tools degrades Climb-WSR while inflating turns (1.28×) and output tokens (1.40×) as agents fall back on brittle scripting workarounds, and frontier models exhibit an over 16× spread in total token usage across multimodal trajectories (from 2.43M to 40.48M tokens per task), with Gemini 4 Carbon matching top deliverable quality at 5.74× lower token usage (6.26× lower API cost) than Claude Fable 5.2 and 2.83× lower token usage than Claude Opus 5.5. Yet while evaluated models exceed 95% on routine regression checks, top Climb-WSR reaches just 56.6%—leaving substantial headroom for agents that natively coordinate visual inspection, code execution, and visual self-verification.

## Acknowledgments

We thank our colleagues across Gemini and the broader Google DeepMind team for their foundational research, infrastructure, and helpful feedback throughout this project. We are especially grateful to the following individuals and partners for their contributions:

- **External Evaluation & Domain Experts:** We thank Artificial Analysis for partnering on standardized harness evaluations (Stirrup) and hosting the public MKA leaderboard, and we thank the domain specialists across all ten verticals whose expertise made this benchmark possible.
- **Initial Autorater Implementation:** Balaji Lakshminarayanan, Xi Liu, and Ben Withbroe.
- **Core Serving & Infrastructure Support:** Steven Guh, Calin Juravle, Nam Luu Nhat, and Sehoon Shon.
- **Program Management Support:** Slavica Andačić, Sumit Bagri, Shruti Garg, Tiffany Loer, Kate Olszewska, and Nandita Sethi.
- **AI Co-Scientist & Editorial Support:** We thank Gemini Argon and Gemini Carbon for serving as AI co-authors, coding collaborators, and editors during the design of the benchmark, analysis of results, and writing of this manuscript.

## References


Anthropic. Claude Opus 4.6 system card. https://www.anthropic.com/claude-opus-4-6-system-card, 2026. URL https://www.anthropic.com/claude-opus-4-6-system-card.

R. K. Arora, J. Wei, R. S. Hicks, P. Bowman, J. Quiñonero-Candela, F. Tsimpourlas, M. Sharman, M. Shah, A. Vallone, A. Beutel, J. Heidecke, and K. Singhal. HealthBench: Evaluating large language models towards improved human health. *arXiv preprint arXiv:2505.08775*, 2025. URL https://arxiv.org/abs/2505.08775.

Artificial Analysis. Stirrup: The lightweight framework for building agents. https://github.com/ArtificialAnalysis/Stirrup, 2025. URL https://github.com/ArtificialAnalysis/Stirrup.

S. Bai, Y. Cai, R. Chen, K. Chen, et al. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025. URL https://arxiv.org/abs/2511.21631.

D. Chen, R. Chen, S. Zhang, Y. Wang, Y. Liu, H. Zhou, Q. Zhang, P. Zhou, Y. Wan, and L. Sun. MLLM-as-a-judge: Assessing multimodal LLM-as-a-judge with vision-language benchmark. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024a. URL https://arxiv.org/abs/2402.04788.

L. Chen, J. Li, X. Dong, P. Zhang, Y. Zang, Z. Chen, H. Duan, J. Wang, Y. Qiao, and D. Lin. Are we on the right way for evaluating large vision-language models? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024b. URL https://arxiv.org/abs/2403.20330.

J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960.

J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5): 378–382, 1971.

X. Fu, Y. Hu, B. Li, Y. Feng, H. Wang, X. Lin, D. Roth, N. A. Smith, W.-C. Ma, and R. Krishna. BLINK: Multimodal large language models can see but not perceive. In *European Conference on Computer Vision (ECCV)*, 2024. URL https://arxiv.org/abs/2404.12390.

S. Garre, E. Ritchie, S. Mehta, and E. Chen. GDP.pdf: Benchmarking grounded multimodal reasoning over professional PDF documents. *arXiv preprint arXiv:2607.11192*, 2026. URL https://arxiv.org/abs/2607.11192.

Google DeepMind. Gemini 3 Pro model card. https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Pro-Model-Card.pdf, 2025. URL https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Pro-Model-Card.pdf.

Google DeepMind. Introducing agentic vision in Gemini 3 Flash. https://blog.google/innovation-and-ai/technology/developers-tools/agentic-vision-gemini-3-flash/, 2026. URL https://blog.google/innovation-and-ai/technology/developers-tools/agentic-vision-gemini-3-flash/.

C. E. Jimenez, J. Yang, A. Wettig, S. Yao, K. Pei, O. Press, and K. Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2310.06770.

K. Li, Z. Meng, H. Lin, Z. Luo, Y. Tian, J. Ma, Z. Huang, and T.-S. Chua. ScreenSpot-Pro: GUI grounding for professional high-resolution computer use. *arXiv preprint arXiv:2504.07981*, 2025a. URL https://arxiv.org/abs/2504.07981.

S. Li, X. Bu, W. Wang, J. Liu, J. Dong, H. He, H. Lu, H. Zhang, C. Jing, Z. Li, et al. MM-BrowseComp: A comprehensive benchmark for multimodal browsing agents. *arXiv preprint arXiv:2508.13186*, 2025b. URL https://arxiv.org/abs/2508.13186.

Y. Ma, Y. Zang, L. Chen, M. Chen, Y. Jiao, X. Li, X. Lu, Z. Liu, Y. Ma, X. Dong, P. Zhang, L. Pan, Y.-G. Jiang, J. Wang, Y. Cao, and A. Sun. MMLongBench-Doc: Benchmarking long-context document understanding with visualizations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024a. URL https://arxiv.org/abs/2407.01523.

Z. Ma, B. Zhang, J. Zhang, J. Yu, X. Zhang, X. Zhang, S. Luo, X. Wang, and J. Tang. SpreadsheetBench: Towards challenging real world spreadsheet manipulation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024b. URL https://arxiv.org/abs/2406.14991.

A. Masry, X. L. Do, J. Q. Tan, S. Joty, and E. Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the Association for Computational Linguistics (ACL)*, 2022. URL https://arxiv.org/abs/2203.10244.

M. Mathew, D. Karatzas, and C. V. Jawahar. DocVQA: A dataset for VQA on document images. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2200–2209, 2021. URL https://arxiv.org/abs/2007.00398.

M. A. Merrill, A. G. Shaw, et al. Terminal-Bench: Benchmarking agents on hard, realistic tasks in command line interfaces. *arXiv preprint arXiv:2601.11868*, 2026. URL https://arxiv.org/abs/2601.11868.

Meta AI. The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation. https://ai.meta.com/blog/llama-4-scout-maverick-behemoth/, 2025. URL https://ai.meta.com/blog/llama-4-scout-maverick-behemoth/.

Meta AI. Introducing Muse Spark: Scaling towards personal superintelligence. https://ai.meta.com/blog/introducing-muse-spark-scaling-towards-personal-superintelligence/, 2026. URL https://ai.meta.com/blog/introducing-muse-spark-scaling-towards-personal-superintelligence/.

G. Mialon, C. Fourrier, C. Swift, T. Wolf, Y. LeCun, and T. Scialom. GAIA: A benchmark for general AI assistants. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2311.12983.

OpenAI. OpenAI o3 and o4-mini system card. https://openai.com/index/o3-o4-mini-system-card/, 2025. URL https://openai.com/index/o3-o4-mini-system-card/.

OpenAI. OpenAI GPT-5 system card. *arXiv preprint arXiv:2601.03267*, 2026. URL https://arxiv.org/abs/2601.03267.

T. Patwardhan, R. Dias, E. Proehl, G. Kim, M. Wang, O. Watkins, S. P. Fishman, M. Aljubeh, P. Thacker, L. Fauconnet, N. S. Kim, P. Chao, S. Miserendino, G. Chabot, D. Li, M. Sharman, A. Barr, A. Glaese, and J. Tworek. GDPval: Evaluating AI model performance on real-world economically valuable tasks. *arXiv preprint arXiv:2510.04374*, 2025. URL https://arxiv.org/abs/2510.04374.

P. Rahmanzadehgervi, L. Bolton, M. R. Taesiri, and A. T. Nguyen. Vision language models are blind. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 18–34, 2024. URL https://arxiv.org/abs/2407.06581.

G. Starace, O. Jaffe, D. Sherburn, J. Aung, J. S. Chan, L. Maksin, R. Dias, E. Mays, B. Kinsella, W. Thompson, J. Heidecke, A. Glaese, and T. Patwardhan. PaperBench: Evaluating AI's ability to replicate AI research. *arXiv preprint arXiv:2504.01848*, 2025. URL https://arxiv.org/abs/2504.01848.

Surge AI. GDP.xlsx: Can agents understand the spreadsheets that run the world? https://huggingface.co/datasets/surgeai/GDP.xlsx, 2026. URL https://huggingface.co/datasets/surgeai/GDP.xlsx.

B. Vidgen, A. Fennelly, E. Pinnix, J. Benchek, B. Foody, et al. The AI productivity index (APEX). *arXiv preprint arXiv:2509.25721*, 2025. URL https://arxiv.org/abs/2509.25721.

B. Vidgen, A. Mann, A. Fennelly, J. W. Stanly, L. Rothman, M. Burstein, J. Benchek, D. Ostrofsky, A. Ravichandran, D. Sur, N. Venugopal, A. Hsia, I. Robinson, C. Huang, O. Varones, D. Khan, M. Haines, A. Bridges, J. Boyle, K. Twist, Z. Richards, C. Mahapatra, B. Foody, and O. Nitski. APEX-Agents. *arXiv preprint arXiv:2601.14242*, 2026. URL https://arxiv.org/abs/2601.14242.

A. Vo, K.-N. Nguyen, M. R. Taesiri, V. T. Dang, A. T. Nguyen, and D. Kim. Vision language models are biased. *arXiv preprint arXiv:2505.23941*, 2025. URL https://arxiv.org/abs/2505.23941.

X. Wang, Y. Chen, L. Yuan, Y. Zhang, Y. Li, H. Peng, and H. Ji. Executable code actions elicit better LLM agents. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024a. URL https://arxiv.org/abs/2402.01030.

Z. Wang, Y. Cui, L. Zhong, Z. Zhang, D. Yin, B. Y. Lin, and J. Shang. OfficeBench: Benchmarking language agents across multiple applications for office automation. *arXiv preprint arXiv:2407.19056*, 2024b. URL https://arxiv.org/abs/2407.19056.

Z. Wang, M. Xia, L. He, H. Chen, Y. Liu, R. Zhu, K. Liang, X. Wu, H. Liu, S. Malladi, A. Chevalier, S. Arora, and D. Chen. CharXiv: Charting gaps in realistic chart understanding in multimodal LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024c. URL https://arxiv.org/abs/2406.18521.

J. Wei, Z. Sun, S. Papay, S. McKinney, J. Han, I. Fulford, H. W. Chung, A. T. Passos, W. Fedus, and A. Glaese. BrowseComp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025. URL https://arxiv.org/abs/2504.12516.

T. Xie, D. Zhang, J. Chen, X. Li, S. Zhao, R. Cao, T. J. Hua, Z. Cheng, D. Shin, F. Lei, Y. Liu, Y. Xu, S. Zhou, S. Savarese, C. Xiong, V. Zhong, and T. Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. URL https://arxiv.org/abs/2404.07972.

F. F. Xu, Y. Song, B. Li, Y. Tang, K. Jain, M. Bao, Z. Z. Wang, X. Zhou, Z. Guo, M. Cao, M. Yang, H. Y. Lu, A. Martin, Z. Su, L. Maben, R. Mehta, W. Chi, L. Jang, Y. Xie, S. Zhou, and G. Neubig. TheAgentCompany: Benchmarking LLM agents on consequential real world tasks. *arXiv preprint arXiv:2412.14161*, 2024. URL https://arxiv.org/abs/2412.14161.

J. Yang, C. E. Jimenez, A. L. Zhang, K. Lieret, J. Yang, X. Wu, O. Press, N. Muennighoff, G. Synnaeve, K. R. Narasimhan, D. Yang, S. I. Wang, and O. Press. SWE-bench Multimodal: Do AI systems generalize to visual software domains? In *International Conference on Learning Representations (ICLR)*, 2025. URL https://arxiv.org/abs/2410.03859.

X. Yue, Y. Ni, K. Zhang, T. Zheng, R. Liu, G. Zhang, S. Stevens, D. Jiang, W. Ren, Y. Sun, C. Wei, B. Yu, R. Yuan, R. Sun, M. Yin, B. Zheng, Z. Yang, Y. Liu, W. Huang, H. Sun, Y. Su, and W. Chen. MMMU: A massive multi-discipline multimodal understanding and reasoning benchmark for expert AGI. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024a. URL https://arxiv.org/abs/2311.16502.

X. Yue, T. Zheng, Y. Ni, Y. Wang, K. Zhang, S. Tong, Y. Sun, B. Yu, G. Zhang, H. Sun, Y. Su, W. Chen, and G. Neubig. MMMU-Pro: A more robust multi-discipline multimodal understanding benchmark. *arXiv preprint arXiv:2409.02813*, 2024b. URL https://arxiv.org/abs/2409.02813.

L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, and I. Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. URL https://arxiv.org/abs/2306.05685.

J. Zhu, W. Wang, Z. Chen, Z. Liu, S. Ye, L. Gu, Y. Duan, H. Tian, W. Su, J. Shao, Z. Gao, E. Cui, Y. Cao, Y. Liu, W. Xu, H. Li, J. Wang, H. Lv, D. Chen, S. Li, Y. He, T. Jiang, J. Luo, Y. Wang, C. He, B. Shi, X. Zhang, W. Shao, J. He, Y. Xiong, W. Qu, P. Sun, P. Jiao, L. Wu, K. Zhang, H. Deng, J. Ge, K. Chen, L. Wang, M. Dou, L. Lu, X. Zhu, T. Lu, D. Lin, Y. Qiao, J. Dai, and W. Wang. InternVL3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025. URL https://arxiv.org/abs/2504.10479.

J. Zhu, Y. Zhang, Z. Ma, B. Zhang, A. Schoepf, D. Woloch, P. Y. Wang, G. R. Yang, S. Jacob, S. Nagisetty, A. Chundru, J. Lin, S. Mateega, and J. Zhang. SpreadsheetBench 2: Evaluating agents on end-to-end business spreadsheet workflows. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2026. URL https://arxiv.org/abs/2606.29955.

M. Zhuge, C. Zhao, D. R. Ashley, W. Wang, D. Khizbullin, Y. Xiong, Z. Liu, E. Chang, R. Krishnamoorthi, Y. Tian, Y. Shi, V. Chandra, and J. Schmidhuber. Agent-as-a-Judge: Evaluate agents with agents. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025. URL https://arxiv.org/abs/2410.10934.

# Appendix

## A. Agentic Autorater Prompt

Listing 1 shows the verbatim prompt template used by the Agentic Autorater (Section 3.2). At runtime, the grading harness populates five placeholders:

- `{deliverables_line}`: Comma-separated list of harvested deliverable filenames staged in `/workspace` (or `(none - the agent submitted no files)`).
- `{chat_line}`: Path to the staged transcript of the candidate's final text response when non-empty (or `(none - the agent said nothing in its final message)`).
- `{task_prompt_section}`: The verbatim task prompt given to the candidate agent, prefixed by `TASK PROMPT GIVEN TO THE AGENT:`.
- `{rubrics_content}`: The task's rubric JSON mapping each `"criterion N"` key to its `description` and integer `weight`.
- `{output_warning_section}`: Empty when the candidate submitted at least one deliverable file, or Listing 2 when the candidate responded only in chat.

Calling `submit_verdicts` validates that every rubric criterion has a structured verdict and terminates the grading session.

*Listing 1 | Agentic Autorater prompt template.*

```
You are grading an agent's work on a task. You are given two kinds of evidence:
the files the agent submitted, and the agent's final chat message. Your task is to
verify whether the agent's work satisfies the criteria
provided below.

The files have been written to the directory /workspace in your environment.
Use the code_exec tool for shell commands (Python scripts, LibreOffice,
pdftoppm, ...) and the view_image tool to look at images or rendered pages.
To inspect a PDF or Office file visually, render its pages to PNG first (e.g.
pdftoppm -png -r 150 <file>.pdf <prefix>) and view the PNGs.
IMPORTANT: When viewing several images, view them one at a time, never output
multiple view_image calls in a single response.

STRATEGY GUIDE FOR OFFICE DOCUMENTS:
When an output artifact is a Microsoft Office file (.docx, .pptx, .xlsx), choose
the most effective evaluation strategy (or a combination) based on the specific
rubric criteria. In most cases, START by rendering the document (Option A): the
extraction libraries in Option B routinely fail to "see" content that a human
grader sees (computed cell values, theme colors, text inside tables), which is a
common cause of FALSE NEGATIVES.
Option A (Rendered Evaluation - PREFER THIS for content presence, values,
  colors, and visual layout): Convert the document to PDF, render its pages to PNG,
  and inspect them with the view_image tool. The rendered PDF shows computed cell
values, theme/fill colors,
  text inside tables, charts, and the visual layout that programmatic extraction
  often misses. Use the code_exec tool to run LibreOffice:
    soffice --headless --norestore --nolockcheck --convert-to pdf <filename> --outdir
/workspace
  Then render the PDF to PNG (pdftoppm -png -r 150 <file>.pdf <prefix>) and use
  the view_image tool on the pages.
  For SPREADSHEETS (.xlsx), ALSO export each sheet to CSV, because a wide or long
  table can overflow the PDF page and be silently clipped, and CSV yields exact,
  unrounded values:
    soffice --headless --norestore --nolockcheck --convert-to csv <filename> --outdir
/workspace
```

```
  Caveat for computed values: LibreOffice does not always recalculate formulas
  when converting a programmatically-generated .xlsx that has no cached results,
  so a computed total may render blank. If an expected value is blank, do NOT
  conclude it is missing - also read the formula (openpyxl data_only=False) or
  compute it yourself before deciding.
  For PAGE COUNT criteria, see the PAGE COUNT section below for the correct method per
format.
Option B (Programmatic Evaluation - for metadata, styles, and exact extraction):
  If the criteria require strict metadata verification, style checks, semantic
  structural validation, or exact text/number extraction (e.g., "does it use the
  H1 style", exact formula text, "read massive tables"), write Python scripts
  (e.g., python-docx, python-pptx, openpyxl) or parse the ZIP/XML directly. Watch
  for these library pitfalls that commonly cause FALSE NEGATIVES; never declare a
  value or section "not present" because of them:
  * Formula cells (openpyxl): load_workbook(data_only=True) returns None for a
    formula cell that has no cached result (common in files generated
    programmatically rather than saved by Excel). If totals, aggregates, or summary
    values read as None, also read the file with data_only=False to see the
    formula, and/or render the file (Option A) to obtain the computed values,
    BEFORE concluding the value is missing.
  * Cell colors (openpyxl): theme-based fills are NOT exposed by
    cell.fill.start_color.index (it reports '00000000' or a theme index, not an
    RGB). Do not infer "no color" from that check alone; inspect fill.patternType
    and the theme/tint attributes, or evaluate the color visually via Option A.
  * Table/box text (python-docx): doc.paragraphs does NOT include text inside
    tables, headers, footers, or text boxes. Before concluding a value or answer
    is "not stated", also iterate doc.tables (and each cell's .paragraphs) and
    check headers/footers - answers in a table cell are otherwise easily missed.

EVIDENCE AVAILABLE TO YOU

1. Deliverables submitted by the agent (files the agent chose to produce):
{deliverables_line}

2. The agent's final chat message. This is a TRANSCRIPT OF WHAT THE AGENT SAID,
   NOT a file the agent created. It has been written to a file only so that you
   can read it with your tools. Never treat its existence as evidence that the
   agent produced a document, and never list it as a deliverable:
{chat_line}{task_prompt_section}

Here is the rubrics JSON containing the criteria to verify:

{rubrics_content}

EVALUATION OF DELIVERABLES AND CHAT RESPONSE:
Always include any artifact submitted by the agent in your
evaluation. Based on the task prompt given to the agent, assess whether it is
appropriate to grade the submitted artifact along with the final chat response:
If the task prompt asked for, implied, or is naturally satisfied by creating a
  deliverable (such as a report, policy brief, memo, review, spreadsheet,
  presentation, code file, plain-text analysis, summary document, etc.), grade
  the submitted artifact(s) and chat response together. Substantive content
  delivered in an artifact fully satisfies criteria requiring that information,
  even if the chat response is concise or acts as a transmittal summary.
Do NOT penalize the agent for delivering content in a submitted file rather
  than directly in chat unless the task prompt explicitly and strictly forbade
  creating or attaching files.
If the task prompt explicitly and strictly prohibited creating or attaching
  files (e.g., "Do not create any files", "Output only in console without creating
  files"), submitting a file violates that instruction.
If no files were submitted, grade the criteria against the chat response on
```

```
  its merits.

GENERAL GRADING PRINCIPLES (apply to EVERY criterion):
1. Honor stated tolerances. When a criterion gives an acceptable range or an
   approximate value (e.g. "acceptable range X to Y", "approximately N", "+/- P%"),
   mark it satisfied if the artifact's value falls within that range, even if it is
   not the exact target. Compute and compare the value with a tool rather than
   relying on visual estimation.
2. Before declaring content ABSENT, inspect the whole artifact and rule out
   tool/library quirks (see the spreadsheet pitfalls above). Content may live in a
   later section, another sheet or tab, a merged cell, or a computed/summary cell.
3. PAGE COUNT -- use the correct method for each format:
   - PDF: page count is baked into the file format. Use:
       pdfinfo <file>.pdf | grep Pages
     or python3 -c "import fitz; print(fitz.open('<file>.pdf').page_count)"
   - PPTX: each slide is a separate file inside the ZIP archive. Count them:
       unzip -l <file>.pptx | grep -c 'ppt/slides/slide[0-9]'
     This is 100% reliable regardless of metadata.
   - DOCX: page count is NOT stored in the file -- it is computed at render time.
     Convert to PDF with LibreOffice and count the PDF pages as an ESTIMATE:
       soffice --headless --norestore --nolockcheck --convert-to pdf <file>.docx
--outdir /workspace
       pdfinfo <file>.pdf | grep Pages
     Note: LibreOffice may render slightly more pages than Microsoft Word for the
     same content (different font metrics and line spacing). If the PDF page count
     exceeds the criterion limit by exactly 1 page, treat it as a BORDERLINE PASS
     -- the extra page is likely a rendering difference, not a genuine violation.

{output_warning_section}

After all verifications complete, call the submit_verdicts tool once with exactly one
verdict per criterion. Each verdict has these fields:
"criterion": the criterion key from the rubrics JSON, e.g. "criterion 1"
"justification": string with your reasoning, written before you decide
"satisfied": boolean (true or false), decided from that reasoning
"sources": list of filenames you referenced to verify the criterion
```

*Listing 2 | Conditional `{output_warning_section}` block injected when the candidate submits no files.*

```
IMPORTANT: The agent submitted NO files; it responded only in its final chat
message. Do NOT mark criteria unsatisfied merely because no file exists. Grade
each criterion against the chat message on its merits based on the task prompt:
if the task prompt strictly required a standalone document or file that was not
produced, criteria requiring that document cannot be satisfied; but substantive
content and answers delivered in the chat message must be evaluated fairly.
```

## B. Case Studies: Visual Perception vs. Programmatic Workarounds

To illustrate how Gemini 4 Carbon adapts when `view_image` is disabled—and where programmatic workarounds break down—we examine two representative tasks from the Stirrup vision ablation (Section 4.3). Table 6 summarizes quality and efficiency across both tasks.

*Table 6 | Comparison of Gemini 4 Carbon w/ `view_image` and w/o `view_image` on two representative MKA tasks illustrating partial coding recovery versus perceptual collapse.*

| Case Study & Arm | Climb-WSR | Overall WSR | Input Tok. | Output Tok. | Turns |
|---|---|---|---|---|---|
| 1. CAD Review (w/ `view_image`) | **76.3** | **88.7** | **1.36M** | **64.1k** | **21.7** |
| 1. CAD Review (w/o `view_image`) | 63.3 | 82.4 | 6.21M | 113.9k | 51.0 |
| Δ / Ratio | −13.0 | −6.2 | 4.6× | 1.8× | 2.4× |
| 2. Road Photo (w/ `view_image`) | **86.7** | **96.0** | **4.87M** | **94.0k** | **43.0** |
| 2. Road Photo (w/o `view_image`) | 14.3 | 64.7 | 18.06M | 207.5k | 76.5 |
| Δ / Ratio | −72.4 | −31.3 | 3.7× | 2.2× | 1.8× |

**Case Study 1: Coding workarounds recover most of the score at 2×–5× compute (Engineering).** In this task, the agent evaluates whether a machined part shown only in a raster CAD drawing can be fabricated using a shop's equipment specifications, producing a structured manufacturability report (Figure 8).

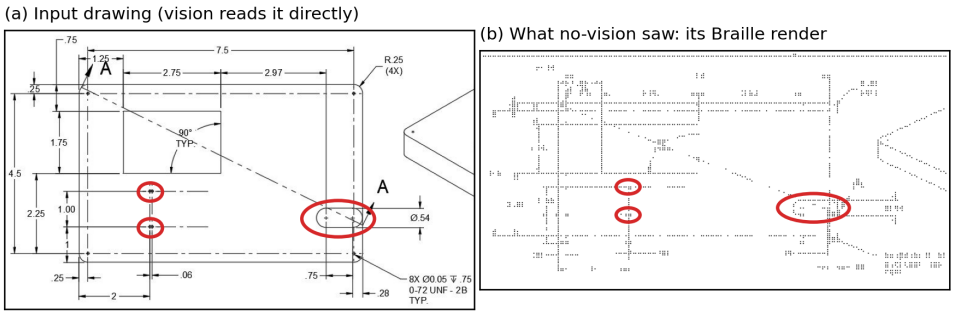



*Figure 8 | **Case Study 1: the workaround recovers most of the score at 2–5× the compute.** (a) The CAD drawing as the agent w/ `view_image` sees it. (b) The Braille render that the agent w/o `view_image` printed from `code_exec`, cropped to the same region. Red rings mark the geometric features that the manufacturability judgment depends on. With `view_image`, the agent catches both in 3/3 runs, while w/o `view_image` catches them in 1/2.*

- **With `view_image`:** Across 23 `view_image` calls on targeted crops of the drawing, the agent directly reads part geometry, tolerances, and feature spacing, consistently flagging non-manufacturable design hazards across all three runs (88.7% Overall WSR, 76.3% Climb-WSR).
- **Without `view_image`:** Thirty of the agent's 49 `code_exec` calls execute custom OpenCV pixel scans alongside ASCII and Unicode Braille renderings of cropped views. By measuring pixel spans between detected edges and pairing them with OCR'd dimension labels, the agent reconstructs an explicit pixel-to-inch scale factor and counts isometric features from Braille dot plots. This effort enables one run to tie the visual baseline's score (86.2% Overall WSR), while the second run misses tight feature-to-wall and hole-to-hole clearance violations (78.7% Overall WSR)—requiring 2.4× the turns, 1.8× the output tokens, and 4.6× the cumulative input tokens.

**Case Study 2: Coding workarounds fail outright despite 2×–4× compute (Construction).** In this task, the agent audits field photographs embedded in a road-inspection PDF against a visual distress classification manual and contractor rate schedules, producing a cost-estimate workbook that must catalog visible pavement defects omitted from the written log (Figure 9).

(a) Report Figure 4 (unlisted pothole circled)



(b) What no-vision saw: its ASCII render



*Figure 9 |* ***Case Study 2: the workaround fails outright despite 2–4× the compute.*** *(a) A site photo from the inspection report. The red ring marks the region that holds the visual evidence the task asks for. (b) The ASCII luminance render that the agent w/o* `view_image` *printed for the same photo. The evidence is present in the render as a dark cluster, but the agent does not recognize it. The agent w/* `view_image` *identifies it in every run. The agent w/o* `view_image` *misses it in all runs, and Climb-WSR drops from 86.7 to 14.3.*

- **With `view_image`:** Using 68 `view_image` calls across extracted inspection photos and zoomed crops, the agent identifies undocumented surface depressions and matches their visual appearance to the corresponding pothole category in the reference manual (96.0% Overall WSR, 86.7% Climb-WSR).
- **Without `view_image`:** Sixty-seven of 84 tool calls run image-processing pipelines—extracting embedded photos, computing OpenCV morphological crack maps, rendering shaded ASCII art, running OCR, and matching ORB keypoints against manual figures. Unable to distinguish three-dimensional bowl depressions from surrounding asphalt texture in ASCII and edge maps, all runs conclude that the relevant figures contain no undocumented distresses and fail every criterion tied to the missing defects. Climb-WSR drops from 86.7% to 14.3% (−72.4 pp) despite 2.2× the output tokens, 1.8× the turns, and 3.7× the input tokens.