MMTB: EVALUATING TERMINAL AGENTS ON MULTIMEDIA-FILE TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Terminals provide a powerful interface for AI agents by exposing diverse tools for automating complex workflows, yet existing terminal-agent benchmarks largely focus on tasks grounded in text, code, and structured files. However, many real-world workflows require practitioners to work directly with audio and video files. Working with such multimedia files calls for terminal agents not only to understand multimedia content, but also to convert auditory and visual evidence across related files into appropriate actions. To evaluate terminal agents on multimedia-file tasks, we introduce MultiMedia-TerminalBench (MMTB), a benchmark of 105 tasks across 5 meta-categories where terminal agents directly operate with audio and video files. Alongside MMTB, we propose Terminus-MM, a multimedia harness that extends Terminus-KIRA with audio and video perception for terminal agents. Together, MMTB and Terminus-MM support a controlled study of multimedia terminal agents, revealing how different forms of multimedia access shape task outcomes and determine which evidence agents rely on to construct executable terminal workflows. MMTB media and metadata are released at https://huggingface.co/datasets/mmtb-anonymous/mmtb-media.

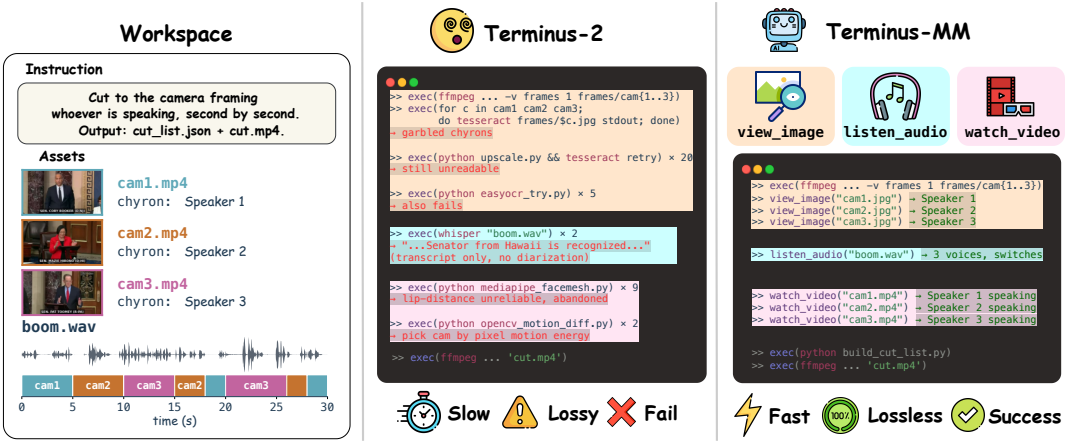


Figure 1: **An example MMTB task and two terminal-agent approaches.** The task merges three videos and one audio file into one edited artifact. Agents with native multimodal access read the raw files directly; text-only agents must reach the same evidence through command-line tools (OCR, ASR, motion-energy), adding processing steps that introduce inefficiency and errors.

1 INTRODUCTION

As terminals provide a powerful interface for AI agents, recent terminal agents such as Claude Code (Anthropic, 2026a) and Codex CLI (OpenAI, 2026) have emerged as practical tools for automating complex command-line workflows. By utilizing shell commands and external tools, terminal agents can interact with files, execute code, search the web, and generate outputs in persistent workspaces. Such capabilities make terminals a natural environment for evaluating realistic

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: **Comparison of MMTB with existing computer-use and audio-visual benchmarks.** *T*, *I*, *A*, and *V* denote text, image, audio, and video, respectively. ▲denotes partial coverage: the benchmark addresses the cross-file aspect but not the audio-visual aspect.

Benchmarks	Input	Content-aware AV reasoning	Cross-file AV reasoning	Persistent file workflow
InterCode (Yang et al., 2023)	T	✗	▲	●
OSWorld (Xie et al., 2024)	$T+I$	✗	▲	●
Terminal-Bench (Merrill et al., 2026)	T	✗	▲	●
OmniBench (Li et al., 2025)	$T+I+A$	●	✗	✗
JointAVBench (Chao et al., 2026)	$T+A+V$	●	✗	✗
AVTrustBench (Chowdhury et al., 2025)	$T+A+V$	●	✗	✗
OmniPlay (Bie et al., 2025)	$T+I+A+V$	●	✗	✗
VideoWebArena (Jang et al., 2025)	$T+I+A+V$	●	✗	✗
MMTB (ours)	$T+I+A+V$	●	●	●

workflows, from planning and tool use to verifiable task completion. Accordingly, recent terminal-agent benchmarks such as Terminal-Bench (Merrill et al., 2026) evaluate terminal agents on diverse realistic workflows using the Harbor task format (Harbor Framework Team, 2026), where each task consists of a user instruction, a working directory, and an expected output specification.

Despite this progress, current terminal-agent benchmarks focus primarily on tasks grounded in text, code, and structured files (Merrill et al., 2026; Yang et al., 2023). However, many real-world workflows require practitioners to work directly with multimedia files such as audio and video recordings. For instance, users may need to prepare media for broadcast or social platforms (Qiu et al., 2024), provide feedback on music or acting performances (Ashutosh et al., 2025), process meetings or compliance-sensitive recordings (Hu et al., 2023), or annotate audio-visual data for research (Gemmeke et al., 2017). Supporting such workflows requires terminal agents to move beyond multimedia understanding alone. They must ground decisions in auditory and visual evidence across files and execute the corresponding actions in a terminal environment. However, existing benchmarks lack multimedia-file tasks designed to evaluate terminal agents (Jang et al., 2025; Li et al., 2025; Merrill et al., 2026).

To this end, we introduce MultiMedia-TerminalBench (MMTB), a benchmark centered on multimedia-file tasks in terminals. As shown in Table 1, MMTB differs from prior computer-use benchmarks, including Terminal-Bench (Merrill et al., 2026), which provide limited coverage of audio and video files and content-aware reasoning over them. It also differs from audio-visual benchmarks focused mainly on multimedia understanding (Bie et al., 2025; Chao et al., 2026; Chowdhury et al., 2025; Li et al., 2025). Specifically, MMTB consists of 105 tasks across 5 meta-categories, with each task grounded in a public source reflecting a paid practitioner workflow, such as those on Upwork or Fiverr websites, and packaged in the Harbor format used by Terminal-Bench (Merrill et al., 2026). Figure 1 further illustrates this design through a workspace example containing a task instruction and the corresponding audio and video files.

To solve the example in Figure 1, conventional terminal agents such as Terminus-2 (Merrill et al., 2026) and Codex CLI (OpenAI, 2026) rely on the intermediate representations of the given multimedia files, rather than directly perceiving their audio or video content. For example, audio may be transformed into spectrograms or RMS signals, while video may be reduced to extracted frames. These additional processing steps can increase the required time and discard important information during conversion. Quantitatively, we observe that the off-the-shelf terminal agent Codex CLI with GPT-5.2 (OpenAI, 2025) solves only 16.2% of MMTB tasks, revealing the limitations of conventional terminal agents on multimedia-file tasks.

To address these limitations, we introduce Terminus-MM, a multimedia terminal-agent harness that extends Terminus-KIRA (KRAFTON AI & Ludo Robotics, 2026) with audio and video perception. In addition, Terminus-MM adapts its perception interface to each workspace by exposing tools matched to the available multimedia files. Using controlled variants that expose different subsets of these perception tools, we analyze how access to each modality affects task outcomes and when agents fall back on command-line conversion.

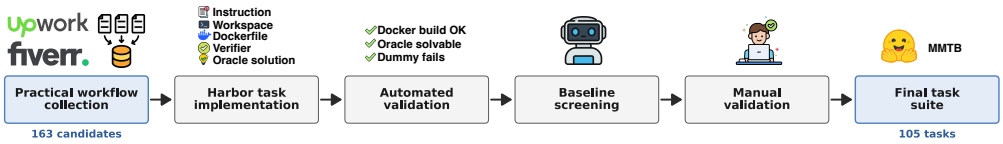


Figure 2: **Construction pipeline of MMTB.** We curate 163 workflow-backed candidate scenarios and adapt them into Harbor tasks with license-compatible substitute multimedia files. Successive automated validation, baseline screening, and manual validation stages revise, refine, and prune the candidates, yielding a final suite of 105 tasks.

Our main contributions are summarized as follows:

1. We introduce MMTB, a benchmark for evaluating terminal agents on multimedia-file tasks, where terminal agents inspect diverse multimedia files, ground terminal actions in multimedia evidence, and produce verifiable output artifacts.
2. Alongside MMTB, we propose Terminus-MM, a multimedia terminal-agent harness extending Terminus-KIRA with audio and video perception, whose interface is adapted to the multimedia files available in each workspace.
3. We ablate native modality access on MMTB with controlled Terminus variants on the same backbone, analyzing how it changes both task success and the way agents gather evidence.

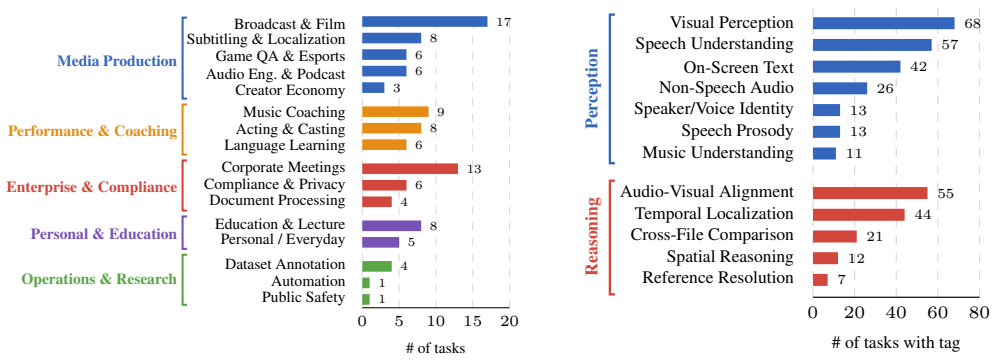
2 MULTIMEDIA-TERMINALBENCH: BENCHMARK DESIGN

In this section, we describe what an MMTB task is, how the 105 tasks were built and filtered, how they are organized and packaged as Harbor tasks, and whether people can solve them.

Benchmark scope. MMTB asks whether agents can carry out the multimedia-file work that practitioners are paid to do, with a particular focus on audio and video files. We pose this work to terminal agents because a terminal gives an agent direct access to the files in its workspace: it can find, inspect, convert, and write them with standard command-line tools. Each task takes place in a persistent terminal workspace whose multimedia assets the agent must turn into a verifiable output. Unlike existing multimedia question-answering benchmarks, where multimedia files typically provide context for answering questions in text, MMTB requires agents to use multimedia evidence to execute terminal actions and produce the required artifact. Existing terminal-agent benchmarks rarely make media the object of work: Terminal-Bench 2.0 (Merrill et al., 2026) includes two video tasks among its 89, one transcribing the on-screen text of a gameplay recording and one locating the take-off and landing frames of a hurdle jump, both of which reduce the video to frames or text and involve no audio.

Benchmark construction pipeline. Figure 2 summarizes the construction pipeline for MMTB. We begin with 163 candidate scenarios, each anchored to a specific public URL documenting a paid practitioner workflow. Most are Upwork and Fiverr gig listings,[1] and the rest are casting calls, practitioner forums, and industry standards documents, so the suite captures the multimedia work people are actually paid to do, not synthetic instruction templates. Each scenario is scoped into a concrete task design, scaffolded as a Harbor task, and populated with license-compatible external media or controlled synthetic and derivative assets. The resulting candidate tasks are then filtered through automated checks for task structure, Docker build, media fetching, oracle solvability, and dummy/no-op failure, baseline screening to catch tasks solvable through shortcuts rather than the intended perception, and manual review of each task and its verifier against the screening trajectories, for unrealistic setups, incorrect verdicts, and redundancy. After filtering, we obtain 105 tasks, each with asset provenance recorded in `media.toml`, including source descriptions, license information, and content hashes. Additional details are provided in Appendices B.1–B.3.

[1] https://www.upwork.com/; https://www.fiverr.com/.



(a) Task categories (b) Capability tags

Figure 3: **Task distribution of MMTB.** (a) Number of tasks in each of the 16 fine-grained workflow categories, grouped into five meta-categories. (b) Number of tasks carrying each capability tag; a task can carry several tags.

Task categories. The final 105 tasks are organized into 5 meta-categories covering practical multimedia workflows, including media production, performance and coaching, enterprise and compliance, personal and education, and operations and research. Within these meta-categories, tasks are further divided into 16 fine-grained workflow categories; Figure 3a gives the number of tasks in each, colored by meta-category. We also annotate each task with multi-label capability tags, which capture the perceptual and reasoning capabilities required by the task; Figure 3b gives the number of tasks carrying each tag, grouped into perception and reasoning tags. Since these tags are not mutually exclusive, their counts are marginal frequencies rather than a partition of the 105 tasks. Frequent tag co-occurrences are reported in Figure 6. The corpus totals 536 media files and 7 h 05 min of timed audio-visual content, with a median per-task duration of 1 min 24 s. Ninety-nine tasks include audio or video; the remaining six are image- or PDF-only workflows. Detailed media-file statistics and an analysis of how agent performance varies with task duration are in Appendix B.6.

Task format. Beyond this suite-level organization, each task is implemented as a Harbor task unit following Terminal-Bench (Merrill et al., 2026), with five components: (i) an *instruction*, which states the user's goal, the required deliverable, and its path and format, without exposing the answer; (ii) a *workspace*, which provides a containerized filesystem with the stored multimedia files and optional supporting files; (iii) an *environment* (Dockerfile), which fixes the container image and preinstalled tools; (iv) a *verifier*, which scores the produced artifact; and (v) an *oracle solution*, which shows that the verifier can be satisfied. Following the Harbor protocol, each task is scored by evaluating the final artifact submitted at the required path, rather than the agent's rationale or command trace. The expected artifact varies by task and may take the form of a selected file, a timestamp or interval, a structured JSON/CSV record, an edit list, or a processed media file. The Harbor format makes each task a self-contained unit that can be run with any terminal agent while the workspace and verifier stay fixed.

Human solvability. To check that tasks can be solved from their instruction and media alone, not only that their verifiers can be satisfied by an oracle solution, we ran a pilot on 40 tasks that we judged feasible for non-experts (Appendix B.10). Seven non-author volunteers without professional media-production background received the same instructions as the agents and produced the deliverables with GUI software of their choice, with no time limit. They passed 70.0% of the tasks, with a median of 14 minutes per task. On the same tasks, the strongest agent configuration (Gemini-3.1-Pro with Terminus-MM) passes 30.0% within its 10-minute budget, and a budget sweep on Gemini-2.5-Flash in Appendix B.9 shows that a longer budget does not improve agent success.

3 Evaluation Setup: Harnesses, Models, Protocol, and Metrics

This section describes how we evaluate terminal agents on MMTB. Our experiments ask two questions: how an agent's ability to carry out multimedia-file tasks changes with the modalities it can perceive

natively, which we study with controlled harnesses built on a minimal terminal agent, and how well strong off-the-shelf terminal agents without native audio or video access handle the same tasks (Section 3.1). Section 3.2 gives the execution protocol shared by all runs, and Section 3.3 the success and cost metrics. The harness and evaluation code are available online[2].

3.1 Harnesses and Models

Controlled harnesses for modality ablation. We want to know how an agent's ability to carry out multimedia-file tasks changes with the modalities it can perceive natively. To isolate this, we start from Terminus-2 (Merrill et al., 2026), a minimal harness in which the agent issues Bash commands, observes their output, and manipulates the filesystem, and add native perception tools to it one modality at a time. Terminus-KIRA (KRAFTON AI & Ludo Robotics, 2026), which adds native image access to Terminus-2, is the text-image member of this family; the other variants add native audio and video access, separately and together (Table 2).

All variants share the same terminal loop and differ only in which native perception tools they expose. A native perception tool passes a workspace file to the backbone through the model's own input channel for that modality; without it, the model sees the file only after shell commands convert it into text or an image. The variants in Table 2 thus form a modality ablation over one backbone's input channels. Terminus-MM exposes all three tools and, for each task, masks the audio or video tool when the workspace has no file of that type, and all three when it has no image, audio, or video file. For the controlled Terminus variants, we use five model backbones: Qwen3.5-122B (Qwen Team, 2026b) and GPT-5.2 (OpenAI, 2025) for the text and image settings, and Gemini-2.5-Flash (Comanici et al., 2025), Gemini-3.1-Pro (Google DeepMind, 2026), and Qwen3.5-Omni-Flash (Qwen Team, 2026a), which support the audio and video ablations.

Table 2: **Controlled Terminus harnesses.** Text, Image, Audio, and Video mark which modalities each harness lets the model perceive natively.

Harness	Text	Image	Audio	Video
Terminus-2	●	✗	✗	✗
Terminus-KIRA	●	●	✗	✗
Terminus-IA	●	●	●	✗
Terminus-IV	●	●	✗	●
Terminus-MM	●	●	●	●

Off-the-shelf terminal agents without native audio or video. The controlled harnesses vary perception while holding the agent loop fixed; we also ask the converse question: how, and how well, do strong off-the-shelf terminal-agent harnesses handle multimedia-file tasks when their backbones lack native audio and video access? Codex CLI (OpenAI, 2026) with GPT-5.2 and Claude Code (Anthropic, 2026a) with Sonnet-4.6 (Anthropic, 2026b) are mature command-line agents with their own agent loops, tool interfaces, and media handling, whose backbones accept text and images but not audio or video. We run them unmodified under the same benchmark protocol, so their results show what strong terminal skills alone achieve on this work and which tasks yield to command-line conversion. Harness implementations and model settings are detailed in Appendix A.

3.2 Execution Protocol

To compare harness and model configurations under a common setting, all agents are evaluated in the same Harbor-style task environment. Each task provides an instruction, a persistent filesystem containing multimedia files, and one or more task-specified output paths. The agent operates within this workspace and must write the required output file or files to the specified paths. The evaluator reads the submitted output files and scores their contents and derived metadata.

Across all agents, the benchmark fixes the task workspace, preinstalled terminal tools, task instructions, verifiers, logging protocol, and 10-minute interaction budget (Appendix B.9 varies it from 5 to 20 minutes). Agents may use ordinary terminal tools such as `ffmpeg`, `ffprobe`, speech transcription, OCR, silence detection, and generated signal-processing scripts to inspect multimedia files indirectly when native perception is unavailable. We allow these command-line workflows since they reflect realistic multimedia-workflow behavior. At the same time, we log commands, perception calls, intermediate files, and final artifacts, allowing the analysis to distinguish runs that use native media access from runs that inspect multimedia files through command-line workflows.

[2] https://anonymous.4open.science/r/mmtb-iclr-anon-2277/.

Table 3: **MMTB results across harnesses and model backbones.** Gemini and Qwen3.5-Omni-Flash rows report mean ± SD (and mean cost and time) over three seeds; other rows are single runs. **Bold** and underline mark the best and second-best on each backbone with audio and video input. T, I, A, V: text, image, audio, video access.

Harness	Model	Modality Access	Success Rate ↑ Binary	Success Rate ↑ Partial	Cost ↓ API	Cost ↓ Time
Terminus-2	Qwen3.5-122B	T	0.105	0.159	$0.101	510s
Terminus-KIRA		T+I	0.095	0.165	$0.233	519s
Terminus-2	GPT-5.2	T	0.105	0.149	$0.335	500s
Terminus-KIRA		T+I	0.114	0.150	$0.645	540s
Terminus-2	Qwen3.5-Omni-Flash	T	$0.025_{\pm 0.005}$	$0.063_{\pm 0.016}$	**$0.079**	570s
Terminus-KIRA		T+I	$0.032_{\pm 0.011}$	$0.048_{\pm 0.012}$	$0.083	592s
Terminus-IA		T+I+A	$0.044_{\pm 0.011}$	$0.078_{\pm 0.017}$	$0.084	572s
Terminus-IV		T+I+V	$0.070_{\pm 0.020}$	$0.113_{\pm 0.031}$	$0.089	547s
Terminus-MM		T+I+A+V	**0.089**$_{\pm 0.005}$	**0.124**$_{\pm 0.010}$	$0.120	**501s**
Terminus-2	Gemini-2.5-Flash	T	$0.060_{\pm 0.005}$	$0.130_{\pm 0.006}$	**$0.190**	205s
Terminus-KIRA		T+I	$0.073_{\pm 0.011}$	$0.128_{\pm 0.001}$	$0.526	281s
Terminus-IA		T+I+A	$0.127_{\pm 0.020}$	$0.196_{\pm 0.037}$	$0.424	228s
Terminus-IV		T+I+V	$0.194_{\pm 0.027}$	$0.265_{\pm 0.038}$	$0.327	209s
Terminus-MM		T+I+A+V	**0.200**$_{\pm 0.025}$	**0.275**$_{\pm 0.030}$	$0.284	**175s**
Terminus-2	Gemini-3.1-Pro	T	$0.156_{\pm 0.033}$	$0.188_{\pm 0.029}$	**$0.787**	523s
Terminus-KIRA		T+I	$0.127_{\pm 0.031}$	$0.162_{\pm 0.023}$	$4.500	531s
Terminus-IA		T+I+A	$0.337_{\pm 0.024}$	$0.426_{\pm 0.043}$	$2.760	393s
Terminus-IV		T+I+V	$0.314_{\pm 0.050}$	$0.445_{\pm 0.018}$	$1.890	**368s**
Terminus-MM		T+I+A+V	**0.340**$_{\pm 0.033}$	**0.458**$_{\pm 0.012}$	$1.901	386s
Claude Code	Sonnet-4.6	T+I	0.162	0.186	$1.735	516s
Codex CLI	GPT-5.2	T+I	0.162	0.202	$2.532	529s

3.3 METRICS

Success metrics. Let T_i denote the i-th task among N tasks, and let $A = (H, M)$ denote an agent configuration consisting of harness H and language model M. Running agent A on T_i transforms the initial workspace into a final workspace state y_i, which includes the generated outputs. A task-specific verifier V_i then evaluates this final state and assigns a partial score $s_i = V_i(y_i)$, where $s_i \in [0, 1]$. Here, evaluation depends only on the final state y_i, not on the agent's intermediate actions, reasoning, or trajectory. Across the task set $\mathcal{T} = \{T_i\}_{i=1}^{N}$, we compute the binary success rate and partial success rate as follows:

$$\textsc{Binary}(A; \mathcal{T}) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left[s_i \geq \tau_i\right], \qquad \textsc{Partial}(A; \mathcal{T}) = \frac{1}{N} \sum_{i=1}^{N} s_i, \tag{1}$$

where τ_i is the task-specific acceptance threshold, set in each task's verifier; for example, `adr-edit-detection` scores interval F1 and passes at $\tau_i = 0.75$. Together, BINARY and PARTIAL evaluate agent configuration A by measuring the fraction of tasks that pass the verifier threshold and the average partial correctness, respectively.

Cost metrics. In addition to success metrics, we report mean API cost per task and mean agent execution time per task. These metrics capture the practical efficiency of each agent configuration, since terminal agents must not only produce correct outputs but also do so within reasonable cost and time. Details of the cost computation and time boundaries are provided in Appendix A.3.

4 RESULTS AND ANALYSES

4.1 NATIVE MULTIMEDIA ACCESS IMPROVES MULTIMEDIA-FILE TASK SOLVING

Table 3 summarizes the main MMTB results. Text-only and text-image access are insufficient for many multimedia-file tasks: on every backbone, harnesses with native audio or video access

Table 4: **Overhead of inspecting multimedia files through command-line conversion tools.** Each ratio divides the API cost or turns of a partial-modality harness by those of Terminus-MM on Gemini-3.1-Pro, over the n matched tasks described in Section 4.2.

Harness	Modality Access	n	API cost ratio		Turn ratio	
			Avg.	Worst	Avg.	Worst
Terminus-2	T	7	4.12×	26.48×	1.00×	2.71×
Terminus-KIRA	T+I	4	1.63×	3.40×	1.39×	2.00×
Terminus-A	T+A	14	4.38×	19.37×	1.77×	3.83×
Terminus-V	T+V	13	1.84×	11.64×	1.15×	2.17×
Terminus-IA	T+I+A	13	4.20×	30.11×	2.21×	8.10×
Terminus-IV	T+I+V	14	7.72×	42.49×	2.01×	5.88×
Terminus-AV	T+A+V	4	2.00×	6.08×	1.28×	2.06×

outperform both baselines, and full access more than doubles their success rate, a gap well beyond seed variation. Full access is also the strongest setting on all three backbones.

Table 8 extends this comparison to single-modality variants. Audio-only and video-only access each improve over text-only access on both backbones, video more than audio. A key bottleneck in MMTB is thus access to media cues such as speech, sound events, motion, timing boundaries, and audio-visual alignment. In other words, native multimedia-file access is an essential component of effective task solving in MMTB.

4.2 Command-Line Conversions Are Less Efficient Than Native Access

When native perception for a required modality is unavailable, agents inspect media through command-line conversion tools. Table 4 quantifies this overhead on matched co-success cases: tasks solved by both a partial-modality harness and Terminus-MM, where the partial harness lacks a required modality and attempts to recover it through terminal tools. Besides the variants in Table 2, Table 4 includes Terminus-A, -V, and -AV, which expose audio, video, or both without image access. We provide the filtering details in Appendix B.12. We report overhead as ratios of USD API cost and trajectory turns relative to Terminus-MM.

Conversion-heavy successful runs incur substantially higher overhead: average API-cost ratios range from 1.63× to 7.72×, with worst cases reaching 30.11× when native video is missing and 42.49× when native access to audio files is missing. Turn ratios also increase in most settings, with a worst case of 8.10×. Because failed and timed-out conversion attempts are excluded, these ratios characterize the cost of successful conversions.

This overhead stems from a longer evidence-acquisition path: the agent must choose an intermediate representation, run the corresponding tools, interpret lossy derived evidence, and often retry before producing the final artifact. Native access shortens this path by letting the agent inspect raw media directly, while terminal commands remain necessary for artifact construction.

4.3 Multimedia Terminal Harnesses Need Both Native Access and Tool-Use Ability

Figure 4 shows that Terminus-MM (Gemini-3.1-Pro) and Codex CLI (GPT-5.2) solve overlapping but non-nested task sets across the 105-task suite: 28 tasks are solved only by Terminus-MM, 6 only by Codex CLI, 11 by both, and 60 by neither. The Terminus-MM-only and Codex-CLI-only task sets suggest different strengths of the two systems. Terminus-MM-only tasks tend to require native modality understanding: the agent listens to audio or watches video directly and grounds its answer in that perceptual evidence. Codex-CLI-only tasks tend to be cases where command-line conversion tools are sufficient: the tools turn media into text or numeric evidence that the agent can reason over. This suggests that a dedicated harness for multimedia-file tasks benefits from both native access and tool-use ability. Appendix B.11 enumerates the detailed partition.

Which perception tools the harness exposes can also matter. Table 5 compares Terminus-MM against Terminus-MM w/o modality masking. Removing the mask lowers both binary and partial success on

Table 5: **Ablation study over modality masking.** We compare Terminus-MM with and without modality masking on Gemini-2.5-Flash and Gemini-3.1-Pro. Bold marks the higher binary and partial success rate on each backbone.

Backbone	Harness	Binary Success Rate ↑	Partial Success Rate ↑
Gemini-2.5-Flash	Terminus-MM w/o modality masking	0.171	0.267
	Terminus-MM	**0.229**	**0.305**
Gemini-3.1-Pro	Terminus-MM w/o modality masking	0.324	0.426
	Terminus-MM	**0.371**	**0.469**

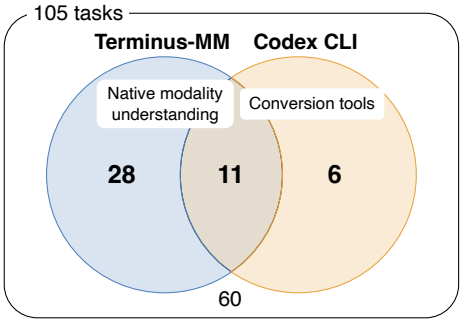


Figure 4: **Overlap of solved tasks across Terminus-MM and Codex CLI.** The non-overlapping regions indicate task subsets for which different capabilities are useful for successful task completion.

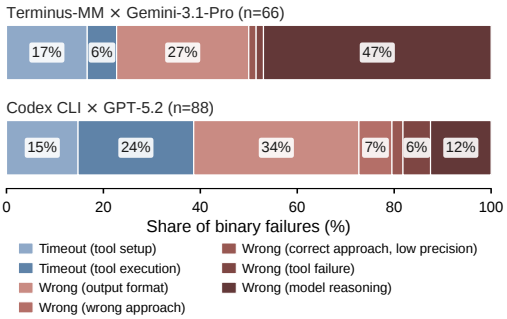


Figure 5: **Failure signatures for failed runs of Terminus-MM and Codex CLI.** Percentages are normalized over failed runs for each agent.

each backbone, suggesting that an unmasked tool list can draw the agent into redundant evidence gathering over unnecessary modalities.

4.4 Failure Modes Differ Between Terminus-MM and Codex CLI

Failure analysis shows that Terminus-MM and Codex CLI fail in different parts of the workflow (Figure 5). We label each of the 66 Terminus-MM and 88 Codex CLI binary-failed runs with its primary failure signature, using an LLM-based classifier whose decision rule we refined by manually inspecting examples (definitions in Appendix B.13). For Terminus-MM, 47.0% of failed runs end with the model misinterpreting or misusing available evidence, and tool-operation failures (tool setup, tool execution, tool failure, and low precision) account for 25.8%. For Codex CLI, which reaches audio and video only through command-line conversion tools, tool-operation failures account for 46.6%, and setup and execution timeouts alone for 38.6%.

Without native audio and video perception, the agent must take the longer evidence-acquisition path of Section 4.2, and each added step is another point of failure before the final artifact is written. Closing the gap requires agents that combine native multimedia access, reliable terminal-tool use, and stronger reasoning over the evidence they gather.

5 Related Work

Multimodal perception benchmarks. MMMU, Video-MME, Video-Holmes, OmniBench, JointAVBench, and AVTrustBench (Yue et al., 2024; Fu et al., 2025; Cheng et al., 2025; Li et al., 2025; Chao et al., 2026; Chowdhury et al., 2025) evaluate image, video, and audio-visual understanding through question answering: the media is placed in the model's context, and the answer is scored against a reference. MMTB draws on the same perceptual skills, such as locating events in time, identifying speakers, and aligning audio with video, but the media arrive as files in a workspace: the agent must locate them, inspect and manipulate them in the terminal, and deliver an output file that a verifier checks (Table 1).

Computer-use and terminal-agent benchmarks. WebArena, VisualWebArena, OSWorld, InterCode, Terminal-Bench, and TerminalWorld (Zhou et al., 2024; Koh et al., 2024; Xie et al., 2024; Yang et al., 2023; Merrill et al., 2026; Chu et al., 2026) evaluate agents that act in web, desktop, or shell environments, largely by checking the environment state they leave behind, but their tasks are grounded mainly in text, code, web pages, and screenshots. VideoWebArena (Jang et al., 2025) gives web agents tutorial videos from which to learn a skill or retrieve a fact, and OmniPlay (Bie et al., 2025) has models play games from visual and auditory observations. In both, the media inform the agent's actions in a web page or game; in MMTB, the media files are what the agent works on, and the deliverable is a file produced from them.

Tool-augmented perception. Another line of work gives language models perception through tools: ViperGPT and VisProg (Surís et al., 2023; Gupta & Kembhavi, 2023) generate programs that call vision modules, and VideoAgent (Wang et al., 2024) answers questions about long videos by querying vision-language models as tools. Terminal agents take the same route with command-line tools, converting audio or video into text, frames, or signal features. In concurrent work, AgenticVBench (Cao et al., 2026) evaluates agents without native audio and video input, and Chen et al. (2026) compare such agents with models that take all input files directly into their context, without a workspace. MMTB instead ablates native modality access on the same terminal tasks and backbone, measuring how it affects both what agents solve and how they gather evidence.

Artifact-level evaluation and benchmark hygiene. SWE-bench, MLE-bench, and AppWorld (Jimenez et al., 2024; Chan et al., 2025; Trivedi et al., 2024) established artifact-level scoring but on code, models, and app state rather than stored media. MMTB sits at the intersection of these lines: agents work on stored multimedia files in a terminal and are scored on the final workspace state they produce. Unlike prior work on benchmark hygiene, such as VQA shortcut analyses (Goyal et al., 2017) and vision-language contamination studies (Chen et al., 2024), MMTB treats an agent's command-line workarounds for a missing modality as evidence about the workflow rather than contamination to suppress; only shortcuts that bypass the media content, such as reading file metadata, are screened out during construction.

6 Conclusion

We introduced MMTB, a rigorously validated benchmark of 105 realistic multimedia-file tasks in persistent terminal workspaces, together with Terminus-MM, a workspace-aware harness that enables native access to audio and video files. MMTB is grounded in practical practitioner workflows, packaged as self-contained Harbor tasks, and filtered through automated checks, oracle solvability tests, baseline screening, and manual validation. Our experiments show that native multimedia access improves agents' ability to complete these practical workflows, while terminal-only approaches often rely on longer and costlier command-line evidence-gathering pipelines. We hope MMTB encourages future terminal agents to perceive audio and video natively; in our ablation, native access complements command-line tools rather than replacing them.

Limitations and extensions. MMTB covers multimedia work whose outcome can be checked deterministically from the produced artifact, such as structured records, annotations, and specified edits; open-ended creative production is outside its scope because such work has no deterministic evaluation. Our findings concern terminal agents, and whether they extend to other agents and multimedia workflows is untested. A natural next step is GUI and desktop agents: such an agent could be evaluated on the same tasks and verifiers once the containerized workspace is ported to a graphical environment. Likewise, the native-access ablation extends to new backbones that accept audio and video input: adding one requires only a thin adaptation layer to the provider's interface, as we did for the Gemini and Qwen endpoints.

AI use statement

In this work, we used generative AI tools to generate synthetic data sets, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, and interpret results. We have not used generative AI tools to help develop theoretical models or conceptual frameworks or to propose

or refine hypotheses, and the tasks to formulate mathematical claims, provide critical ingredients for proving mathematical claims, and assist in the writing of proofs are not applicable to this work. Additionally, we used generative AI tools for brainstorming and to identify relevant literature, summarize or analyse existing literature, create or modify scientific figures or images, create or edit software code, draft parts of a research paper, edit a research paper to improve readability, and format references. We have reviewed all AI-assisted work: the authors verified AI-generated code, data, and text, and all reported success rates are computed by programmatic verifiers from logged agent runs. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics statement

All media in MMTB are sourced under licenses that permit redistribution for research or are synthesized, with per-asset provenance and license records released alongside the tasks (Appendices B.1 and B.2). The human pilot in Appendix B.10 involved seven adult volunteers who are not authors of this work; they gave informed consent and participated without compensation. The pilot was informal and did not undergo institutional ethics review; it recorded only pseudonymous identifiers, submitted task outputs, and completion times, and collected no personal or sensitive information.

Reproducibility statement

All 105 tasks are released in the Harbor task format with their instructions, environments, reference solutions, and artifact-level verifiers, and the media files, with per-asset provenance, licenses, and content hashes, are available at https://huggingface.co/datasets/mmtb-anonymous/mmtb-media. The harness and evaluation code is available at https://anonymous.4open.science/r/mmtb-iclr-anon-2277/. Harness implementations, model identifiers and routes, and the cost and time measurement are specified in Appendices A.1, A.2, and A.3.

References

Anthropic. Claude Code. https://code.claude.com/docs/en/overview, 2026a. Accessed: 2026-05-06. 1, 5

Anthropic. Introducing Claude Sonnet 4.6. https://www.anthropic.com/news/claude-sonnet-4-6, 2026b. Accessed: 2026-05-06. 5

Kumar Ashutosh, Tushar Nagarajan, Georgios Pavlakos, Kris Kitani, and Kristen Grauman. ExpertAF: Expert actionable feedback from video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 2

Fuqing Bie, Shiyu Huang, Xijia Tao, Zhiqin Fang, Leyi Pan, Junzhe Chen, Min Ren, Liuyu Xiang, and Zhaofeng He. OmniPlay: Benchmarking omni-modal models on omni-modal game playing. *arXiv preprint arXiv:2508.04361*, 2025. 2, 9

Zongheng Cao, Yi Zheng, Rui Song, and Xinyu Hu. AgenticVBench: Can AI agents complete real-world post-production tasks? *arXiv preprint arXiv:2605.27705*, 2026. 9

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Lilian Weng, and Aleksander Mądry. MLE-bench: Evaluating machine learning agents on machine learning engineering. In *International Conference on Learning Representations (ICLR)*, 2025. 9

Jianghan Chao, Jianzhang Gao, Wenhui Tan, Yuchong Sun, Ruihua Song, and Liyun Ru. JointAVBench: A benchmark for joint audio-visual reasoning evaluation. In *International Conference on Learning Representations (ICLR)*, 2026. 2, 8

Dongping Chen, Xuanao Huang, Zhihan Hu, Qingyuan Shi, Dianqi Li, and Tianyi Zhou. Sandboxed coding agents are competitive omni-modal task solvers. *arXiv preprint arXiv:2606.00579*, 2026. 9

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are we on the right way for evaluating large vision-language models? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 9

Junhao Cheng, Yuying Ge, Teng Wang, Yixiao Ge, Jing Liao, and Ying Shan. Video-Holmes: Can MLLM think like Holmes for complex video reasoning? *arXiv preprint arXiv:2505.21374*, 2025. 8

Sanjoy Chowdhury, Sayan Nag, Subhrajyoti Dasgupta, Yaoting Wang, Mohamed Elhoseiny, Ruohan Gao, and Dinesh Manocha. AVTrustBench: Assessing and enhancing reliability and robustness in audio-visual LLMs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. 2, 8

Zhaoyang Chu, Jiarui Hu, Xingyu Jiang, Pengyu Zou, Han Li, Chao Peng, Peter O'Hearn, Earl T. Barr, Mark Harman, Federica Sarro, and He Ye. TerminalWorld: Benchmarking agents on real-world terminal tasks. *arXiv preprint arXiv:2605.22535*, 2026. 9

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. 5

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, Peixian Chen, Yanwei Li, Shaohui Lin, Sirui Zhao, Ke Li, Tong Xu, Xiawu Zheng, Enhong Chen, Caifeng Shan, Ran He, and Xing Sun. Video-MME: The first-ever comprehensive evaluation benchmark of multi-modal LLMs in video analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 8

Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio Set: An ontology and human-labeled dataset for audio events. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017. 2

Google DeepMind. Gemini 3.1 Pro — Model Card. https://deepmind.google/models/model-cards/gemini-3-1-pro/, 2026. Accessed: 2026-05-06. 5

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 9

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 9

Harbor Framework Team. Harbor: A framework for evaluating and optimizing agents and models in container environments. https://github.com/harbor-framework/harbor, 2026. doi:10.5281/zenodo.20953922. Accessed: 2026-09-19. 2

Yebowen Hu, Timothy Ganter, Hanieh Deilamsalehy, Franck Dernoncourt, Hassan Foroosh, and Fei Liu. MeetingBank: A benchmark dataset for meeting summarization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023. 2

Lawrence Keunho Jang, Yinheng Li, Dan Zhao, Charles Ding, Justin Lin, Paul Pu Liang, Rogerio Bonatti, and Kazuhito Koishida. VideoWebArena: Evaluating long context multimodal agents with video understanding web tasks. In *International Conference on Learning Representations (ICLR)*, 2025. 2, 9

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations (ICLR)*, 2024. 9

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Russ Salakhutdinov, and Daniel Fried. VisualWebArena: Evaluating multimodal agents on realistic visual web tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024. 9

KRAFTON AI and Ludo Robotics. Terminus-KIRA: Boosting frontier model performance on Terminal-Bench with minimal harness. https://github.com/krafton-ai/KIRA, 2026. Accessed: 2026-09-19. 2, 5

Yizhi Li, Yinghao Ma, Ge Zhang, Ruibin Yuan, King Zhu, Hangyu Guo, Yiming Liang, Jiaheng Liu, Zekun Moore Wang, Jian Yang, et al. OmniBench: Towards the future of universal omni-language models. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2025. 2, 8

Mike A. Merrill, Alexander G. Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E. Kelly Buchanan, et al. Terminal-Bench: Benchmarking agents on hard, realistic tasks in command line interfaces. In *International Conference on Learning Representations (ICLR)*, 2026. 2, 3, 4, 5, 9

OpenAI. Update to GPT-5 system card: GPT-5.2. https://openai.com/index/gpt-5-system-card-update-gpt-5-2/, 2025. Accessed: 2026-05-06. 2, 5

OpenAI. Codex CLI. https://learn.chatgpt.com/docs/codex/cli, 2026. Accessed: 2026-09-19. 1, 2, 5

Jielin Qiu, Jiacheng Zhu, William Han, Aditesh Kumar, Karthik Mittal, Claire Jin, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Ding Zhao, Bo Li, and Lijuan Wang. MMSum: A dataset for multimodal summarization and thumbnail generation of videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2

Qwen Team. Qwen3.5-Omni technical report. *arXiv preprint arXiv:2604.15804*, 2026a. 5

Qwen Team. Qwen3.5: Towards native multimodal agents. https://qwen.ai/blog?id=qwen3.5, 2026b. Accessed: 2026-09-19. 5

Dídac Surís, Sachit Menon, and Carl Vondrick. ViperGPT: Visual inference via Python execution for reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 9

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A controllable world of apps and people for benchmarking interactive coding agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024. 9

Xiaohan Wang, Yuhui Zhang, Orr Zohar, and Serena Yeung-Levy. VideoAgent: Long-form video understanding with large language model as agent. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 9

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2024. 2, 9

John Yang, Akshara Prabhakar, Karthik Narasimhan, and Shunyu Yao. InterCode: Standardizing and benchmarking interactive coding with execution feedback. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2023. 2, 9

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu 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)*, 2024. 8

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations (ICLR)*, 2024. 9

A Implementation Details

A.1 Harness Implementation Details

This appendix supports Section 3.1. Table 6 summarizes the nine Terminus-family harnesses evaluated in the paper, listing their native perception tools and masking policy. Algorithm 1 states the workspace-aware modality masking used by Terminus-MM; the remaining harnesses use a static schema. Terminus-2 is Harbor's built-in implementation and Terminus-KIRA runs its vendored prompt; the variants we build share the Terminus-MM code and prompt skeleton and differ only in the exposed tools. `watch_video` passes a video file to the model together with its soundtrack, and `listen_audio` passes standalone audio files. Terminus-KIRA uses more tokens than Terminus-2 on four of the five backbones (Table 8), which accounts for its higher cost.

Table 6: **Harness implementations evaluated in this paper.** Modality masking is *static* (the perception schema is fixed at construction) or *dynamic* (the schema is re-derived per task at run start from a workspace scan).

Harness	Perception tools	Masking
Terminus-2	—	static
Terminus-KIRA	view_image	static
Terminus-A	listen_audio	static
Terminus-V	watch_video	static
Terminus-AV	listen_audio, watch_video	static
Terminus-IA	view_image, listen_audio	static
Terminus-IV	view_image, watch_video	static
Terminus-MM w/o modality masking	view_image, listen_audio, watch_video	static
Terminus-MM	subset of {view_image, listen_audio, watch_video}	dynamic

Workspace-aware modality masking (Terminus-MM). The *static* harnesses fix their perception schema at construction time. Terminus-MM instead derives the schema once per task at run start by scanning the workspace, mapping file extensions to perception modalities, and keeping only the perception tools whose target modality is present in the workspace (Algorithm 1). The `view_image` keep rule is "include `view_image` whenever any media file is present", even on audio-only or video-only workspaces, so that frames or spectrograms produced by `execute_commands` retain a visual perception path. The three tasks whose only inputs are PDF documents receive no perception tool.

Algorithm 1 Workspace-aware modality masking (Terminus-MM). Invoked once per task at run start, before the first LLM call. The tool schema sent to the LLM is the masked subset returned here.

files ← ListFiles(workspace_dir, max_depth=6)
modalities ← ∅
for $f \in$ files **do**
    ext ← Extension(f)
    if ext ∈ {.wav,.mp3,.ogg,.flac,.aac,.m4a} **then** modalities ← modalities ∪ {audio}
    end if
    if ext ∈ {.mp4,.webm,.avi,.mov,.mkv} **then** modalities ← modalities ∪ {video}
    end if
    if ext ∈ {.png,.jpg,.jpeg,.gif,.webp} **then** modalities ← modalities ∪ {image}
    end if
end for
keep ← {execute_commands, task_complete}
if modalities $\neq \emptyset$ **then** keep ← keep ∪ {view_image}
end if
if audio ∈ modalities **then** keep ← keep ∪ {listen_audio}
end if
if video ∈ modalities **then** keep ← keep ∪ {watch_video}
end if
return $\{t \in \text{MM_TOOLS} : t.\text{name} \in \text{keep}\}$

Prompt-template structure. The MMTB prompt templates share a fixed skeleton: a system-role preamble, a per-tool description block, an agent-constraints block (no human intervention; minimal state changes before `task_complete`), and `{instruction}` plus `{terminal_state}` placeholders that Harbor formats at task start. The variants differ only in which per-tool description blocks are present. The full Terminus-MM template lists all five tools (`execute_commands`, `task_complete`, `view_image`, `watch_video`, `listen_audio`) and is used by Terminus-MM with and without modality masking. Terminus-IA, -IV, -A, -V, and -AV drop the blocks of their missing tools and add a one-line "You CANNOT call ..." disclaimer. Terminus-KIRA uses a structurally distinct Apache-2.0-vendored prompt that predates the MM family.

A.2 Backbone Model Details

Model names in Table 3 are abbreviations chosen for table density. Full identifiers, provider routes, and version slugs for every model that appears in the paper are in Table 7. The full per-cell results table corresponding to the abbreviated main-text version is extended in Table 8. All Terminus runs use sampling temperature 0.7 (the Harbor default); Codex CLI and Claude Code use their product defaults.

Table 7: **Backbone model details.** The "In paper" column gives the abbreviation used in the main-text tables. The "Full identifier" column gives the model name as listed by the provider. The "Input" column gives the modalities the model accepts (T, I, A, V: text, image, audio, video). The "Route / version" column gives the access path used for the reported runs (provider slug routed via OpenRouter, native API, or installable terminal agent). Runs queried between `2026-04-29` and `2026-07-26` (UTC).

In paper	Full identifier	Route / version	Input	Type
Controlled-harness backbones				
Qwen3.5-122B	Qwen3.5-122B-A10B	`qwen/qwen3.5-122b-a10b` via OpenRouter	T+I	open
GPT-5.2	OpenAI GPT-5.2	`openai/gpt-5.2` via OpenRouter	T+I	closed
Gemini-2.5-Flash	Google Gemini 2.5 Flash	`google/gemini-2.5-flash` via OpenRouter	T+I+A+V	closed
Gemini-3.1-Pro	Google Gemini 3.1 Pro Preview	`google/gemini-3.1-pro-preview` via OpenRouter	T+I+A+V	closed
Qwen3.5-Omni-Flash	Qwen3.5-Omni-Flash	`qwen3.5-omni-flash-2026-03-15` via DashScope	T+I+A+V	closed
Installable CLI-agent backbones (subscription-routed)				
Sonnet-4.6	Anthropic Claude Sonnet 4.6	`anthropic/claude-sonnet-4-6` via Claude Code (Claude Max OAuth)	T+I	closed
GPT-5.2	OpenAI GPT-5.2	`openai/gpt-5.2` via Codex CLI (Codex Pro OAuth)	T+I	closed

A.3 Cost and Time Measurement Methodology

This appendix specifies the API cost and time bases used in Table 3 and in the cost ratios reported in Table 4 (the cost of inspecting a missing modality through command-line tools, relative to native perception on the same task).

API cost. We report a uniform-proxy cost computed identically across all harnesses as $\text{cost}_{\text{uniform}} = n_{\text{input}} \times r_{\text{input}} + n_{\text{output}} \times r_{\text{output}}$, where r_{input} and r_{output} are the posted per-token list rates for the model and n_{input} is the total prompt tokens including any cached portion (no prompt-cache discount applied). We chose this uniform basis because prompt-cache token capture was asymmetric across harnesses in our sweep: cached-token counts were recorded for Terminus-2 but not for the other Terminus harnesses on Gemini routes. Reporting cache-discounted billed cost would mix real billed amounts (Terminus-2 only) with token-proxy estimates without discount (other harnesses); the uniform-proxy basis treats every row as un-cached at the same rate. Absolute USD figures should be read as un-cached upper bounds; agents that rely heavily on prompt caching at deployment (notably Codex CLI) will incur proportionally less in production. Posted rates are recorded in the released cost-aggregation script (snapshot 2026-04-29). All costs use these fixed list rates rather than the rates in effect when each run was made (May 2026 for seed 1, July 2026 for seeds 2 and 3). The seed-2 and seed-3 Gemini

Table 8: **Modality ablation across backbones.** Rows with ± are three-seed means (success rate as mean ± SD; cost, tokens, and time averaged over the seeds), as in Table 3; rows without ± are single runs. Each Terminus row exposes a different subset of perception tools to the agent: T = text-only shell; I = `view_image`; A = `listen_audio`; V = `watch_video`. Terminus-MM's schema is masked per workspace: it drops perception tools whose target file types are absent from the workspace at run start (Appendix A.1); the remaining Terminus rows use a static schema. Outcomes report mean binary success and mean task-specific partial credit. Cost columns: API = mean USD per task; Tokens = mean total tokens per task (input + cache + output, in thousands); Time = mean agent execution wall (excludes container setup and verifier scoring).

Harness	Model	Access	Success Rate ↑		Cost ↓		
			Binary	Partial	API	Tokens	Time
Terminus-2	Qwen3.5-122B	T	0.105	0.159	$0.101	302.7k	510s
Terminus-KIRA		T+I	0.095	0.165	$0.233	725.8k	519s
Terminus-2	GPT-5.2	T	0.105	0.149	$0.335	94.2k	500s
Terminus-KIRA		T+I	0.114	0.150	$0.645	248.6k	540s
Terminus-2	Qwen3.5-Omni-Flash	T	$0.025_{\pm 0.005}$	$0.063_{\pm 0.016}$	$0.079	170.4k	570s
Terminus-KIRA		T+I	$0.032_{\pm 0.011}$	$0.048_{\pm 0.012}$	$0.083	169.7k	592s
Terminus-IA		T+I+A	$0.044_{\pm 0.011}$	$0.078_{\pm 0.017}$	$0.084	167.6k	572s
Terminus-IV		T+I+V	$0.070_{\pm 0.020}$	$0.113_{\pm 0.031}$	$0.089	185.8k	547s
Terminus-MM		T+I+A+V	$0.089_{\pm 0.005}$	$0.124_{\pm 0.010}$	$0.120	247.0k	501s
Terminus-2	Gemini-2.5-Flash	T	$0.060_{\pm 0.005}$	$0.130_{\pm 0.006}$	$0.190	562.3k	205s
Terminus-KIRA		T+I	$0.073_{\pm 0.011}$	$0.128_{\pm 0.001}$	$0.526	1,630.6k	281s
Terminus-A		T+A	0.105	0.183	$0.127	359.7k	209s
Terminus-V		T+V	0.229	0.307	$0.140	411.8k	210s
Terminus-AV		T+A+V	0.219	0.315	$0.139	406.8k	188s
Terminus-IA		T+I+A	$0.127_{\pm 0.020}$	$0.196_{\pm 0.037}$	$0.424	1,323.3k	228s
Terminus-IV		T+I+V	$0.194_{\pm 0.027}$	$0.265_{\pm 0.038}$	$0.327	1,014.0k	209s
Terminus-MM		T+I+A+V	$0.200_{\pm 0.025}$	$0.275_{\pm 0.030}$	$0.284	881.8k	175s
Terminus-2	Gemini-3.1-Pro	T	$0.156_{\pm 0.033}$	$0.188_{\pm 0.029}$	$0.787	275.3k	523s
Terminus-KIRA		T+I	$0.127_{\pm 0.031}$	$0.162_{\pm 0.023}$	$4.500	2,154.3k	531s
Terminus-A		T+A	0.257	0.349	$1.992	904.9k	493s
Terminus-V		T+V	0.286	0.395	$1.118	481.5k	417s
Terminus-AV		T+A+V	0.276	0.408	$1.029	442.1k	419s
Terminus-IA		T+I+A	$0.337_{\pm 0.024}$	$0.426_{\pm 0.043}$	$2.760	1,292.1k	393s
Terminus-IV		T+I+V	$0.314_{\pm 0.050}$	$0.445_{\pm 0.018}$	$1.890	858.8k	368s
Terminus-MM		T+I+A+V	$0.340_{\pm 0.033}$	$0.458_{\pm 0.012}$	$1.901	864.7k	386s
Claude Code	Sonnet-4.6	T+I	0.162	0.186	$1.735	516.3k	516s
Codex CLI	GPT-5.2	T+I	0.162	0.202	$2.532	1,364.6k	529s

runs used more input tokens per task than seed 1 in every configuration (1.1 to 6.0 times), so costs computed from seed 1 alone (Table 4 and the appendix breakdowns) are lower than the three-seed means in Table 3. For Qwen3.5-Omni-Flash, served through Alibaba Cloud Model Studio, we apply its posted International rates (price list accessed on 2026-09-18, after the runs): $0.40 per million text, image, and video input tokens, $3.00 per million audio input tokens, and $2.20 or $11.90 per million output tokens for requests with text-only or multimodal input, respectively, computed per request from logged token usage and scaled to each run's token totals. On this backbone Terminus-MM is the costliest harness (Table 3) for two reasons visible in the logged usage: video and image inputs add roughly 38k input tokens per task, and the outputs of every request that carries audio, image, or video input are billed at the higher rate, so Terminus-MM's output cost is about 1.6× that of Terminus-2 despite fewer output tokens.

Time. We report the agent execution wall, measured by Harbor's `TimingInfo` wrapper around the agent's run entry point in the Harbor trial runner. This window includes LLM API latency, tool-call execution, and perception-tool latency. It excludes container or sandbox setup, harness initialization, and verifier scoring. Time boundaries are identical across all evaluated harnesses because they share the same `Trial` wrapper code path.

A.4 Compute Resources

The single-seed Terminus sweeps ran on Daytona managed sandboxes with inference routed through OpenRouter; the seed replications, the budget sweep, and all Qwen3.5-Omni-Flash runs ran in local Docker containers, with Qwen3.5-Omni-Flash served through DashScope. Codex CLI and Claude Code are run in a local Docker container with OAuth subscription authentication. All inference is API-mediated; no local GPU is used. Task containers have internet access, which agents use to install tools at run time.

The single-seed grid of 2,520 task runs (24 model–harness configurations $\times$ 105 tasks) takes approximately 30 wall-clock hours when sandboxes are dispatched in parallel. These 24 are the 20 Terminus rows of Table 8 on Qwen3.5-122B, GPT-5.2, Gemini-2.5-Flash, and Gemini-3.1-Pro, Terminus-MM w/o modality masking on the two Gemini backbones, Codex CLI, and Claude Code. The three-seed replication adds 20 Gemini runs (two extra seeds for ten configurations) and 15 Qwen3.5-Omni-Flash runs (three seeds for five configurations), and the budget sweep adds 6 runs, each over the 105 tasks. The per-trajectory analyses in Sections 4.2–4.4 and the appendix breakdowns use the seed-1 run of each cell, the original run made before the replication. We additionally ran preliminary or superseded experiments (replaced baselines, harness ablation variants) that do not appear in the reported tables.

B Supplementary Analyses

B.1 Public Release, License, and Croissant Metadata

The benchmark is publicly hosted on Hugging Face as a single dataset mirror linked from the supplementary material. Per-asset media licenses are recorded in each task's `media.toml` (open licenses for source media, with per-license counts in Table 10; MIT for benchmark code; see Appendix B.2 for the full breakdown including a small number of CC-BY-NC and GPL files covered by the license caveats below) and propagate into the dataset's Croissant 1.1 metadata, which we hand-craft to include six fields from the Croissant RAI vocabulary and `prov:wasGeneratedBy` (`rai:dataLimitations`, `rai:dataBiases`, `rai:personalSensitiveInformation`, `rai:dataUseCases`, `rai:dataSocialImpact`, `rai:hasSyntheticData`, `prov:wasGeneratedBy`) plus Croissant-RAI extensions for collection, preprocessing, annotation, and maintenance protocols. The Croissant file passes all four checks of a public Croissant validator (JSON format, Croissant schema, record generation, RAI completeness). A Gebru-style `DATASHEET.md` ships alongside the dataset and covers motivation, composition, collection, preprocessing, uses, distribution, and maintenance in the standard seven-section narrative. Per-task self-contained subdirectories (1 MB to $\sim$300 MB of media each) serve as natural samples for reviewers without requiring the full $\sim$6 GB download. The solution and verifier scripts of every task carry a fixed canary string so that their appearance in training corpora can be detected.

B.2 Source Corpus, Language Coverage, Demographics

Source media combine real recordings under open licenses (NASA archives, archive.org, Wikimedia Commons, Blender open movies, CREMA-D, LibriSpeech) with synthesized speech and assets, many of them layered on real recordings. Synthetic content is used where ground truth must be controlled, from targeted inserts such as a planted offset to fully synthetic assets such as TTS dialogues, rendered UI screen recordings, and game footage. The 105 tasks are predominantly English-speech; bilingual material appears in localization-flavored tasks (German and French in document subsets, dubbed content in subtitling tasks). Systematic coverage of non-Latin scripts and right-to-left languages is not yet provided. Speakers and on-camera actors are sourced from publicly licensed media; a per-task demographic balance audit is planned but not yet delivered. Out-of-scope by construction: live streaming, real-time interactive media, very long-form content (>1 h), audio-less gameplay capture, and embodied / robotic-camera footage.

Synthesis manifest. Synthetic assets are generated with the tools listed in Table 9 (full per-asset provenance lives in `media.toml` and the Croissant `rai:machineAnnotationTools` field).

Aggregate license breakdown. Table 10 gives license counts across the 536 media records declared in the tasks' `media.toml` files, grouped by license family.

Table 9: **Synthesis-tooling manifest.** Per-asset provenance is recorded in each task's `media.toml`.

Tool	Use
`ffmpeg`	audio/video processing, defect injection
FluidSynth + FluidR3_GM SoundFont	MIDI → audio rendering
LilyPond 2.26.0	music notation engraving (Bach, Mozart, etc.)
Kokoro-82M (Apache-2.0)	speech synthesis
Piper TTS	speech synthesis (four tasks)
Godot 4.x + Kenney (CC0)	gameplay-QA footage rendering
Wav2Lip	lip-sync video synthesis (one task)
`reportlab` / `wkhtmltopdf`	PDF document synthesis
`matplotlib`	diagram rendering (educational tasks)
`music21`	MIDI extraction from public-domain scores
Playwright / Chromium + Pillow	UI screen recordings, image rendering
custom `build_assets.py`	per-task deterministic synthesis scripts

Table 10: **Aggregate license breakdown for source files across the 105-task suite.** Counts derive from the per-asset license fields recorded in each task's `media.toml`; non-commercial and GPL-family files are addressed in the license caveats below.

License family	N files	Examples
CC-BY family (2.0 / 3.0 / 4.0, incl. SA)	170	Blender open movies, Wikimedia Commons, LibriSpeech, author-contributed footage
Apache-2.0 (incl. Kokoro-82M TTS)	94	TTS-synthesized speech
Public Domain	47	Historical recordings, NASA
ODbL-1.0	88	CREMA-D actor speech recordings
MIT	80	Author-synthesized assets
CC0-1.0 (incl. Kenney)	38	Game graphics + SFX
GPL family	7	Footage in `game-alert-mismatch` (see caveats)
Two licenses (MIT / Apache-2.0; PD / CC-BY-SA)	6	Game footage, mixed caption sources
CC-BY-NC-*	3	Three lecture-clip tasks (see caveats)
GFDL 1.2	2	Wikimedia legacy
Wav2Lip-derived (non-commercial research use)	1	Lip-synced panel in `debate-attribution` (see caveats)
Total	536	

License caveats. One task (`debate-attribution`) includes a video lip-synced with Wav2Lip, whose non-commercial research-use license applies to that file. Three lecture-content tasks (`chapter-repair`, `lecture-demo-clip-extract`, `long-form-clip-miner`) include lecture recordings (MIT Missing Semester and a PyCon AU talk) under CC-BY-NC-SA-4.0 or CC-BY-NC-4.0; redistribution by users for commercial purposes requires separate licensing. One game-QA task (`game-alert-mismatch`) includes seven footage clips from GPL-licensed open-source games (six under GPL-3.0+, one under GPL-2.0+); downstream users redistributing that task's media bytes are subject to the GPL's share-alike obligations on the bytes themselves (the surrounding verifier and oracle code are unaffected). Both license families permit academic-research use, which is the benchmark's intended purpose. Per-file licenses are recorded in each task's `media.toml` and in the Croissant metadata.

B.3 Task Selection and Screening

Of the 163 candidate scenarios, 105 were released. A candidate was dropped when a synthetic asset was not realistic enough, when it duplicated an accepted task, or when no suitable media could be obtained.

Every released task must be solved by perceiving the information it places in the media. We tested this with construction-time runs: we ran Terminus-KIRA and Terminus-MM on Gemini-3.1-Pro on each task as it was built and read every trajectory for shortcuts that bypass that information, such as

reading file metadata instead of the content. Passing or failing was not itself a criterion: the released suite contains 11 tasks that Terminus-KIRA passed at construction time and 70 that Terminus-MM failed.

Every verifier was checked against these construction-time Terminus-KIRA and Terminus-MM runs. It had to accept the oracle solution, and for each run we read the trajectory alongside the verifier's verdict on the produced artifact, confirming that correct deliverables were accepted and incorrect or shortcut ones rejected. Whenever any check in the pipeline failed, the task was dropped or fixed and passed through the pipeline again. The last task was merged on 2026-04-28; every evaluation run reported in this paper started on or after 2026-04-29.

B.4 Capability-Tag Co-Occurrence

Figure 6 shows frequent tag pairs; Figure 7 and Table 11 break down success and cost by capability tag, and Table 12 by meta-category.

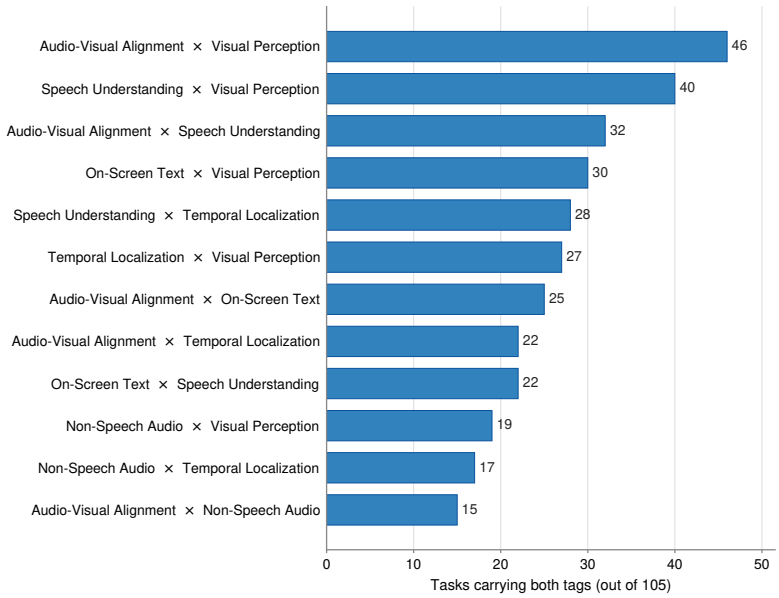


Figure 6: **Top capability-tag co-occurrence pairs across the 105-task suite under the twelve capability tags.** Each bar counts the tasks that carry both tags. Strongest pairs are Audio-Visual Alignment × Visual Perception (46 tasks), Speech Understanding × Visual Perception (40), and Audio-Visual Alignment × Speech Understanding (32), reflecting the multimodal-grounding emphasis of the suite.

B.5 Task Inventory by Fine-Grained Category

The five meta-categories used throughout the paper (Table 12, Table 13) are defined as follows. **Media Production** (n=40): tasks that produce or quality-check media artifacts for an external audience, spanning subtitling and localization, broadcast and film post-production, podcast assembly, game-capture review, and social-media clip mining. **Performance & Coaching** (n=23): tasks that evaluate a third party's executed performance, such as an actor's take, a language learner's pronunciation, or a musician's rendition, and localize what was done correctly or incorrectly. **Enterprise & Compliance** (n=23): workplace tasks that extract decisions or audit evidence from meeting and screen-share recordings, or that structure business documents such as receipts, invoices, and reports. **Personal & Education** (n=13): tasks operating on the user's own media or learning materials, including personal and family recordings and lecture content consumed for self-study. **Operations & Research** (n=6): operational and research workflows without an external audience or performer, including ML dataset annotation, smart-device automation, and public-safety surveillance audits.

Table 11: **Per-capability-tag success-rate and cost breakdown for the seed-1 run of every backbone × harness cell of Table 3 except Qwen3.5-Omni-Flash.** Costs use the uniform proxy of Appendix A.3; because they come from the seed-1 run, Gemini costs differ from the three-seed means in Table 3. Capability tags are *multi-label*: a task carrying k tags contributes to all k columns. Sum of n across tag columns is 369 (103 of 105 tasks carry ≥ 2 tags; max 6 tags/task). Per-cell metrics are unweighted means over the tasks bearing that tag. Per-tag column triplet reports binary success rate (B), partial success rate (P), and mean USD cost ($). **Bold** marks the best harness within the Flash or Pro Terminus family (5 cells per family); underline marks the 2nd-best within the same family. Gold highlights the global #1 cell across all 16 backbone×harness rows for each metric column; silver highlights the global #2.

Backbone	Harness	Audio-Visual Alignment (n=55)			Cross-File Comparison (n=21)			Music Understanding (n=11)			Non-Speech Audio (n=26)			On-Screen Text (n=42)			Reference Resolution (n=7)			Spatial Reasoning (n=12)			Speaker/Voice Identity (n=13)			Speech Prosody (n=13)			Speech Understanding (n=57)			Temporal Localization (n=44)			Visual Perception (n=68)		
		B	P	$	B	P	$	B	P	$	B	P	$	B	P	$	B	P	$	B	P	$	B	P	$	B	P	$	B	P	$	B	P	$	B	P	$
Qwen3.5-122B	Terminus-2	0.036	0.062	$0.113	0.191	0.189	$0.098	0.000	0.000	$0.158	0.115	0.167	$0.116	0.095	0.118	$0.112	0.000	0.000	$0.088	0.167	0.202	$0.139	0.077	0.171	$0.134	0.154	0.280	$0.069	0.088	0.140	$0.083	0.114	0.195	$0.089	0.088	0.137	$0.108
	Terminus-KIRA	0.054	0.126	$0.244	0.095	0.121	$0.206	0.000	0.057	$0.199	0.115	0.211	$0.257	0.119	0.170	$0.232	0.000	0.000	$0.254	0.333	0.352	$0.256	0.000	0.143	$0.184	0.077	0.138	$0.248	0.035	0.096	$0.231	0.114	0.174	$0.255	0.118	0.195	$0.248
GPT-5.2	Terminus-2	0.073	0.099	$0.334	0.238	0.275	$0.395	0.091	0.091	$0.390	0.115	0.202	$0.345	0.071	0.076	$0.332	0.000	0.000	$0.308	0.083	0.110	$0.276	0.077	0.138	$0.376	0.077	0.171	$0.444	0.053	0.114	$0.296	0.114	0.170	$0.314	0.118	0.150	$0.340
	Terminus-KIRA	0.073	0.082	$0.670	0.048	0.067	$0.696	0.000	0.015	$0.569	0.154	0.209	$0.624	0.119	0.122	$0.693	0.143	0.143	$0.741	0.167	0.191	$0.606	0.077	0.133	$0.568	0.000	0.090	$0.685	0.105	0.127	$0.683	0.182	0.213	$0.621	0.147	0.151	$0.680
Gemini-2.5-Flash	Terminus-2	0.091	0.128	**$0.097**	0.143	0.159	$0.099	0.000	0.043	$0.155	0.038	0.158	**$0.085**	0.000	0.065	$0.172	0.000	0.000	$0.076	0.083	0.111	$0.172	0.077	0.103	**$0.096**	**0.154**	**0.253**	**$0.044**	0.070	0.133	$0.113	0.068	0.179	**$0.137**	0.073	0.137	$0.114
	Terminus-KIRA	0.054	0.105	$0.280	0.048	0.114	$0.127	**0.091**	**0.144**	$0.215	0.077	0.152	$0.181	0.071	0.136	$0.300	0.000	0.000	$0.322	0.167	0.222	$0.242	0.077	0.148	$0.233	0.000	0.035	$0.133	0.035	0.100	$0.282	0.068	0.115	$0.261	0.088	0.149	$0.269
	Terminus-IA	0.054	0.119	$0.276	0.191	0.204	$0.159	0.000	0.000	$0.261	0.077	0.149	$0.223	0.191	0.214	$0.334	0.143	0.167	$0.050	0.250	0.281	$0.205	0.154	0.237	$0.145	0.077	0.120	$0.059	0.123	0.178	$0.237	0.091	0.141	$0.324	0.176	0.230	$0.232
	Terminus-IV	0.182	0.258	$0.187	0.095	0.138	$0.166	0.000	0.029	$0.205	**0.115**	0.201	$0.135	0.191	0.202	$0.182	0.286	0.417	**$0.015**	0.167	0.171	$0.261	**0.308**	**0.357**	$0.180	0.000	0.058	$0.191	0.088	0.221	$0.146	0.114	0.155	$0.190	0.235	0.301	$0.196
	Terminus-MM	**0.255**	**0.320**	$0.102	**0.238**	**0.263**	**$0.070**	0.091	0.091	**$0.096**	0.112	**0.211**	$0.143	**0.262**	**0.289**	**$0.093**	**0.429**	**0.441**	$0.020	**0.333**	**0.390**	**$0.061**	0.154	0.243	$0.130	0.077	0.230	$0.073	**0.316**	**0.388**	**$0.079**	**0.250**	**0.355**	$0.140	**0.309**	**0.375**	**$0.089**
Gemini-3.1-Pro	Terminus-2	0.091	0.097	**$0.779**	0.191	0.191	$0.818	0.000	0.000	$1.072	0.231	0.269	**$0.859**	0.119	0.124	**$0.741**	0.000	0.000	**$0.540**	0.250	0.285	**$0.459**	0.154	0.210	**$0.805**	0.077	0.212	$1.020	0.053	0.096	**$0.686**	0.159	0.174	**$0.727**	0.162	0.179	**$0.685**
	Terminus-KIRA	0.054	0.067	$2.176	0.143	0.179	$2.198	0.091	0.109	**$1.019**	0.115	0.192	$2.308	0.143	0.156	$1.990	0.000	0.071	$2.233	0.250	0.321	$1.538	0.077	0.173	$1.576	0.077	0.220	$2.226	0.035	0.105	$2.335	0.114	0.160	$2.214	0.118	0.135	$2.178
	Terminus-IA	0.291	0.362	$2.261	**0.476**	**0.490**	$1.385	0.091	0.102	$1.030	0.269	0.341	$2.205	**0.357**	**0.440**	$1.818	0.429	0.509	$2.111	0.500	0.588	$1.961	0.231	0.322	$1.380	**0.385**	**0.368**	**$0.780**	0.298	0.402	$1.943	0.250	0.329	$1.944	0.382	0.442	$2.081
	Terminus-IV	0.255	0.417	$1.314	0.333	0.374	$0.927	0.000	0.089	$1.396	0.269	0.378	$1.311	0.333	0.413	$1.314	**0.571**	0.607	$1.495	**0.583**	**0.664**	$0.952	0.231	0.264	$1.703	0.308	**0.403**	$1.024	0.368	0.481	$1.302	0.341	**0.426**	$1.440	0.368	0.398	$1.215
	Terminus-MM	**0.327**	**0.454**	$1.336	0.381	0.411	**$0.663**	**0.182**	**0.222**	$1.256	**0.385**	**0.457**	$1.031	0.357	0.436	$1.351	0.571	**0.655**	$1.028	0.500	0.640	$1.026	**0.308**	**0.501**	$1.667	0.308	0.383	$0.906	**0.386**	**0.519**	$1.471	**0.364**	0.425	$1.334	**0.441**	**0.540**	$1.185
Sonnet-4.6	Claude Code	0.127	0.122	$1.877	0.286	0.287	$1.395	0.091	0.126	$1.523	0.269	0.337	$1.629	0.167	0.173	$1.868	0.143	0.143	$3.161	0.417	0.402	$1.710	0.154	0.151	$1.954	0.077	0.115	$1.315	0.088	0.097	$2.005	0.091	0.127	$1.774	0.235	0.236	$1.957
GPT-5.2	Codex CLI	0.109	0.139	$2.653	0.191	0.191	$2.057	0.091	0.091	$2.861	0.192	0.235	$2.132	0.167	0.218	$2.815	0.143	0.241	$4.208	0.250	0.307	$2.852	0.154	0.154	$2.135	0.000	0.038	$1.861	0.123	0.173	$2.880	0.159	0.211	$2.306	0.221	0.259	$2.741

Table 12: **Per-meta-category success-rate and cost breakdown for the seed-1 run of every backbone × harness cell of Table 3 except Qwen3.5-Omni-Flash.** Costs use the uniform proxy of Appendix A.3; because they come from the seed-1 run, Gemini costs differ from the three-seed means in Table 3. Five meta-categories cover all 105 tasks ($\sum n_{\text{tasks}} = 105$). Per-cell metrics are weighted means over the constituent fine-categories. Each meta-category column triplet reports binary success rate (B), partial success rate (P), and mean USD cost ($). **Bold** marks the best harness within the Flash or Pro Terminus family (5 cells per family); underline marks the 2nd-best within the same family. Gold highlights the global #1 cell across all 16 backbone×harness rows for each metric column; silver highlights the global #2. Definitions of the five meta-categories are given in Appendix B.5.

Backbone	Harness	Media Production (n=40)			Performance & Coaching (n=23)			Enterprise & Compliance (n=23)			Personal & Education (n=13)			Operations & Research (n=6)		
		B	P	$	B	P	$	B	P	$	B	P	$	B	P	$
Qwen3.5-122B	Terminus-2	0.075	0.141	$0.105	0.130	0.196	$0.107	0.130	0.132	$0.104	0.154	0.222	$0.082	0.000	0.104	$0.088
	Terminus-KIRA	0.075	0.195	$0.219	0.043	0.084	$0.234	0.174	0.175	$0.204	0.154	0.205	$0.284	0.000	0.146	$0.317
GPT-5.2	Terminus-2	0.150	0.186	$0.299	0.043	0.097	$0.426	0.087	0.104	$0.306	0.154	0.272	$0.289	0.000	0.000	$0.431
	Terminus-KIRA	0.125	0.184	$0.610	0.000	0.051	$0.637	0.087	0.103	$0.661	0.308	0.294	$0.705	0.167	0.167	$0.721
Gemini-2.5-Flash	Terminus-2	0.075	0.149	$0.136	**0.087**	**0.156**	$0.099	0.000	0.008	$0.145	0.077	0.235	**$0.027**	**0.167**	0.241	**$0.123**
	Terminus-KIRA	0.100	0.168	$0.252	0.000	0.046	$0.177	0.130	0.195	$0.260	0.000	0.051	$0.206	0.000	0.104	$0.150
	Terminus-IA	0.100	0.163	$0.316	0.043	0.068	$0.140	0.261	0.293	$0.193	0.231	0.258	$0.191	0.000	0.140	$0.301
	Terminus-IV	0.100	0.179	$0.188	0.000	0.043	$0.191	**0.391**	**0.456**	$0.115	0.231	0.265	$0.198	0.167	0.208	$0.372
	Terminus-MM	**0.225**	**0.327**	**$0.131**	0.043	0.130	**$0.096**	0.391	0.424	**$0.050**	**0.385**	**0.355**	$0.047	0.000	**0.263**	$0.195
Gemini-3.1-Pro	Terminus-2	0.125	0.172	**$0.734**	0.087	0.157	$1.107	0.130	0.140	**$0.568**	0.154	0.194	**$0.675**	0.167	0.133	**$0.733**
	Terminus-KIRA	0.075	0.127	$2.179	0.087	0.186	$1.777	0.174	0.200	$2.031	0.077	0.154	$2.316	0.167	0.133	$1.931
	Terminus-IA	0.250	0.343	$1.981	0.217	0.286	**$1.032**	0.522	0.582	$1.917	0.385	0.401	$2.040	**0.500**	0.617	$1.562
	Terminus-IV	0.225	0.349	$1.356	0.217	**0.292**	$1.189	**0.565**	0.599	$1.242	**0.462**	**0.536**	$1.379	0.333	**0.659**	$1.109
	Terminus-MM	**0.300**	**0.471**	$1.271	**0.261**	0.290	$1.070	0.565	**0.629**	$1.409	0.385	0.431	$1.028	0.500	0.615	$1.289
Sonnet-4.6	Claude Code	0.150	0.195	$1.640	0.043	0.087	$1.430	0.217	0.211	$2.049	0.308	0.295	$2.178	0.167	0.167	$1.363
GPT-5.2	Codex CLI	0.225	0.258	$2.144	0.000	0.022	$2.306	0.174	0.257	$3.252	0.154	0.154	$2.698	0.333	0.417	$2.869

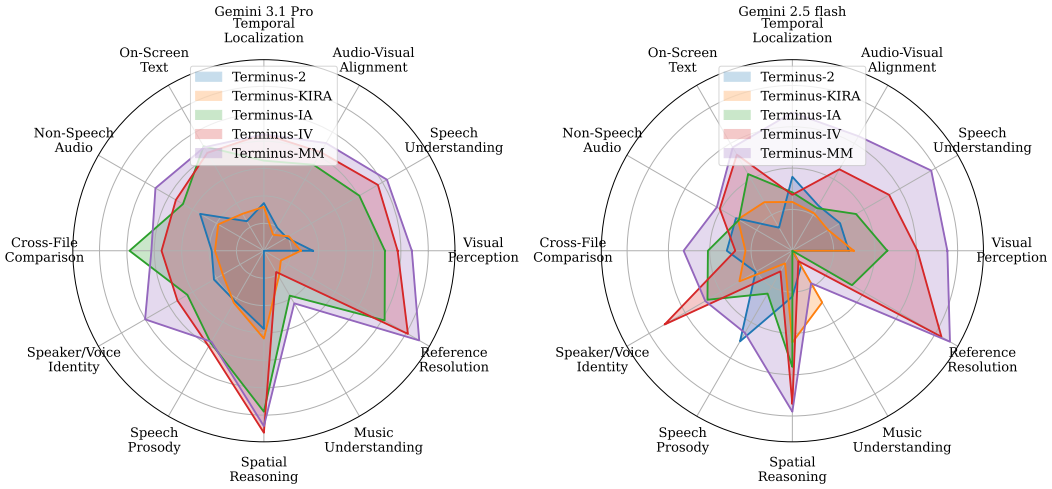


Figure 7: **Partial success rate per capability tag for the five controlled harnesses on Gemini-3.1-Pro and Gemini-2.5-Flash.**

Table 13: **Task counts by fine-grained category, grouped by meta-category.** All 16 categories are populated; totals sum to 105 released tasks across 5 meta-categories.

Meta-category	Fine-grained category	N
Media Production (40)	Broadcast & Film Production	17
	Subtitling & Localization	8
	Game QA & Esports	6
	Audio Engineering & Podcast Production	6
	Creator Economy & Social Media	3
Performance & Coaching (23)	Music Coaching & Performance Feedback	9
	Acting & Casting	8
	Language Learning & Speech Coaching	6
Enterprise & Compliance (23)	Corporate Workflows & Meetings	13
	Compliance, Privacy & Public Release	6
	Document Processing & Bookkeeping	4
Personal & Education (13)	Education & Lecture Content	8
	Personal / Everyday	5
Operations & Research (6)	Dataset & ML Annotation	4
	Automation & Smart Devices	1
	Public Safety & City Ops	1
Total		**105**

B.6 Per-Task Media Volume and Difficulty

We report per-task media-volume statistics as a proxy for the labor a human practitioner would face on each task. All durations are ffprobe-measured against the `sha256`-pinned asset files declared in `media.toml`; image and PDF entries contribute to file counts but not to duration totals.

The mix differs by meta-category: Performance & Coaching is audio-heavy (4.91 audio files / task on average, mostly short music or speech samples), Personal & Education concentrates timed content in long-form lectures (9 min 50 s mean per task), and Operations & Research carries the largest file counts per task (15.33 files, mostly image batches for dataset annotation). The corpus-wide distribution is also skewed: median per-task duration is 1 min 24 s while the mean is 4 min 03 s and the maximum is 48 min 55 s on the longest single-lecture task.

Table 14: **Per-meta-category media-volume statistics (averages per task).** "files" is the mean count of all asset files; "image / video / audio" break that count down by extension (other static formats such as PDF contribute to the file count but are not enumerated). "Avg. duration" sums per-task video and audio duration and averages across the tasks in the meta-category. The corpus totals 536 media files and 7 h 05 min of timed video and audio.

Meta-category	n	files	image	video	audio	Avg. duration
Media Production	40	4.08	0.07	2.93	0.82	5 min 35 s
Performance & Coaching	23	6.13	0.39	0.35	4.91	0 min 52 s
Enterprise & Compliance	23	4.35	0.26	0.96	1.78	1 min 45 s
Personal & Education	13	3.08	0.85	1.69	0.46	9 min 50 s
Operations & Research	6	15.33	8.33	6.00	0.83	2 min 15 s
All tasks	**105**	**5.10**	**0.75**	**1.95**	**1.89**	**4 min 03 s**

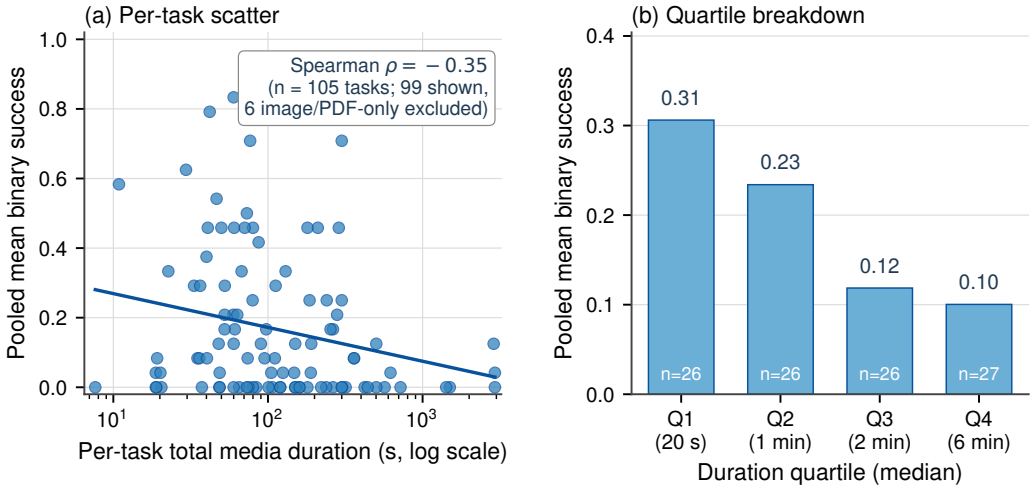


Figure 8: **Per-task media duration vs. pooled mean binary success across the 24 single-seed harness×model configurations (Appendix A.4).** (a) Scatter on log-x with a least-squares fit (Spearman $\rho = -0.35$); the six image- or PDF-only tasks with zero timed media duration are shown in panel (b)'s leftmost quartile but excluded from the log-x scatter. (b) The same data binned into four equal-count duration quartiles: mean binary success drops from 30.6% in the shortest quartile ($\leq$ 46.5 s) to 10.0% in the longest quartile (median 6 min, range 221 s–49 min). The trend is monotonic but moderate; no single duration threshold is the cause of the gap, and file-count quartiles show no comparable trend ($\rho \approx 0$ for files-vs-success, not shown).

B.7 Strategy-Divergence Case Studies

Figure 9 shows case studies of the aggregate access gap at the level of individual trajectories, rather than as an artifact of averaging. Terminus-MM alone solves the meeting-reference task; on take selection, native audio (Terminus-A, Terminus-MM) and a pitch-plot conversion (Terminus-KIRA) both succeed; and the A/V-desync task defeats all four harnesses.

B.8 Domain-Level Modality Patterns (Heatmap)

Figure 10 implies that how much native access matters depends on the workflow type (Table 15 gives the rates pooled over both Gemini backbones). In Enterprise and Compliance, full access raises success from 7% to 48%, the largest gain of any category, so access is the main lever there. In Performance and Coaching, full access leaves it the hardest category regardless (15%, the lowest of the five), so something other than perception access, most likely the fine-grained judgment these tasks require, is the binding constraint.

Task pattern	Terminus-2	Terminus-KIRA	Terminus-A	Terminus-MM
A/V desync detection av-desync-detection	ffprobe stream-metadata heuristics, blind-sort	ffprobe + frame extract, view_image	ffprobe packets + listen_audio samples	watch_video each clip then sort
Voice-over take selection prosody-take-selection	read brief, blind-guess take filename	librosa f0 plot, file-stat heuristic	listen_audio each take, write pick	listen_audio each take, write pick
Meeting decision logging spoken-decision-cell-ref	Whisper STT + frame-OCR pipeline	frame OCR + view_image overrides	listen_audio + frame-OCR + write CSV	watch_video + view_image key frames

Figure 9: **Representative strategy divergence across harnesses on Gemini-3.1-Pro.** Each cell summarizes the dominant per-turn pattern observed in the agent's trajectory; cell color marks whether that harness solved the task (green) or failed (red).

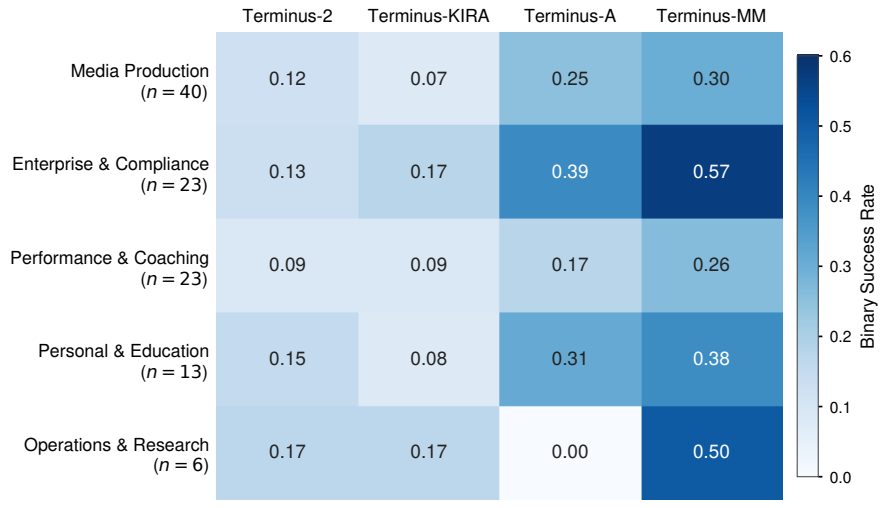


Figure 10: **Per-domain modality dependency on Gemini-3.1-Pro.** Cells show the binary success rate over the tasks in each meta-category. The four controlled harnesses are Terminus-2 (text-only), Terminus-KIRA (text+image), Terminus-A (text+audio), and Terminus-MM (full access).

B.9 Budget Sensitivity

We reran Terminus-2 and Terminus-MM on Gemini-2.5-Flash at 5, 15, and 20 minutes (all 105 tasks, one run per cell; Table 16). Binary success shows no consistent improvement across the fourfold range and the gap between Terminus-MM and Terminus-2 persists at every budget (0.12–0.16); longer

Table 15: **Binary success rate by meta-category for Terminus-2 (text-only), Terminus-KIRA (text+image), and Terminus-MM (full access), pooled across Gemini-2.5-Flash and Gemini-3.1-Pro (seed 1).** Task runs = tasks in the category × 2 backbones.

Category	Task runs	Text-only	Text+image	Full access
Enterprise and Compliance	46	7%	15%	**48%**
Personal and Education	26	12%	4%	**38%**
Media Production	80	10%	9%	**26%**
Operations and Research	12	17%	8%	**25%**
Performance and Coaching	46	9%	4%	**15%**
Total	210	10%	9%	**30%**

budgets mainly add cost as failing runs consume the window. Gemini-3.1-Pro and the CLI agents were not swept.

Table 16: **Budget sweep on Gemini-2.5-Flash: binary / partial success (billed API cost per task, agent seconds).** The 10 min column is the three-seed mean; billed cost differs from the uniform proxy of Table 3.

Harness	5 min	10 min	15 min	20 min
Terminus-2	0.067 / 0.106 ($0.039, 124 s)	0.060 / 0.130 ($0.068, 205 s)	0.057 / 0.120 ($0.077, 228 s)	0.048 / 0.112 ($0.194, 296 s)
Terminus-MM	0.229 / 0.287 ($0.083, 110 s)	0.200 / 0.275 ($0.090, 175 s)	0.181 / 0.244 ($0.143, 177 s)	0.190 / 0.297 ($0.143, 235 s)

B.10 Human Pilot on a 40-Task Subset

We select tasks that an ordinary GUI user could plausibly solve without domain-expert knowledge or professional tools, and run a small human study on that pool: seven non-author participants attempt tasks from the 40-task pool, viewing the media and producing each deliverable with GUI software of their choice, with no terminal access and no time limit. Participants are colleagues without professional media-production background and receive the same instructions as the agents. Table 17 reports the results alongside Terminus-MM on the same 40 tasks, and Table 18 the time to completion. Humans solve these tasks at far higher rates than Gemini-3.1-Pro with Terminus-MM, the strongest agent configuration (70.0% against 30.0% binary success, in each of three seeds); mean time to completion is 18.7 minutes (median 14.1). Participants are non-expert, so this is a solvability reference point rather than a professional baseline, and the 40-task pool, limited to tasks that do not require specialized multimedia-professional skill, is not representative of the full suite.

Table 17: **Human study results and Terminus-MM performance on the same 40 tasks (agent cells: three-seed mean ± SD).**

Method	Binary	Partial
Qwen3.5-Omni-Flash + Terminus-MM	$0.042_{\pm 0.014}$	$0.073_{\pm 0.011}$
Gemini-2.5-Flash + Terminus-MM	$0.142_{\pm 0.029}$	$0.207_{\pm 0.021}$
Gemini-3.1-Pro + Terminus-MM	$0.300_{\pm 0.000}$	$0.405_{\pm 0.038}$
Human (non-expert)	**0.700**	**0.712**

Table 18: **Time to completion in the human study, all 40 tasks.**

Time	Tasks
0–10 min	13
10–20 min	16
20–30 min	3
30–40 min	5
40–50 min	0
50–60 min	1
60+ min	2

Table 19: **Task-level binary-pass partition for Terminus-MM and Codex CLI.**

Regime	Definition	Tasks	Share
Both solve	Codex CLI pass $\wedge$ Terminus-MM pass	11	10.5%
Codex CLI only	Codex CLI pass $\wedge$ Terminus-MM fail	6	5.7%
Terminus-MM only	Codex CLI fail $\wedge$ Terminus-MM pass	28	26.7%
Both fail	Codex CLI fail $\wedge$ Terminus-MM fail	60	57.1%

Table 20: **Metadata signatures of the four solver regimes.**

Regime	Main metadata signals	Interpretation
Both solve	11 tasks; audio+video in 8/11; many JSON/CSV or simple edit outputs	Either system can reach the artifact once the media is reduced to reliable derived signals, such as a transcript, a handful of frames, or a feature table, through ordinary command-line tools.
Codex CLI only	6 tasks; audio+video in 4/6; keywords cluster around OCR, DSP, synchronization, repair, and deterministic media editing	The decisive evidence is accessible through ordinary command-line operations such as transcribing speech, OCR'ing frames, and computing signal features, so strong scripting and artifact repair can beat native perception.
Terminus-MM only	28 tasks; audio+video in 17/28; native audio in 23/28; CSV/JSON outputs in 67.9%	Grounding-limited workflows. Many outputs are per-event records where each row depends on co-grounding audio cues with visual events or states.
Both fail	60 tasks; native audio in 59/60; video in 37/60; joint-A/V keyword in 21/60; JSON outputs in 68.3%	Remaining headroom. The hardest cases combine audio-rich evidence, joint A/V grounding, timing precision, and structured artifact production.

B.11 Regime Analysis for Terminus-MM and Codex CLI

This appendix supports the solver-regime analysis in Section 4.3. We compare task-level binary outcomes between Terminus-MM with Gemini-3.1-Pro and Codex CLI with GPT-5.2. Here, Terminus-MM denotes the full-modality harness with modality masking in its seed-1 run (39 of 105 tasks passed). The comparison is observational: the two systems differ in both backbone and harness, so the partitions should be read as observed solver regimes rather than causal proofs about which modality is strictly necessary.

Partition construction. For each of the 105 tasks, we take the binary-pass outcome for Terminus-MM and Codex CLI and assign the task to one of four regimes: both systems pass, only Codex CLI passes, only Terminus-MM passes, or both systems fail. All 105 tasks are paired; no task is missing from either sweep. Table 19 gives the resulting partition.

Regime interpretation. The four-way split shows that the two systems are not related by strict containment. The Terminus-MM-only set is larger than the Codex-CLI-only set, but the Codex-CLI-only set is non-empty and contains tasks where terminal pipelines outperform the full-modality harness. We therefore interpret the regimes as different bottlenecks. Codex-CLI-only tasks are *pipeline-limited*: the media evidence can be converted into stable intermediate signals and executable edits. Most Terminus-MM-only tasks are *grounding-limited*: the output depends on matching audio evidence to visual events, screen states, or temporally localized actions. Two Terminus-MM-only tasks (`blood-test-pdfs-to-csv`, `invoice-estimate-pdfs-to-xlsx`) take only PDF inputs, for which Terminus-MM exposes no perception tool. Both-fail tasks are *combined-bottleneck* cases: solving them requires both precise media grounding and robust terminal artifact construction.

Task lists for the solved and disagreement regimes. Table 21 lists the tasks in each regime. Figure 1 depicts `multicam-active-speaker-cut`, a both-fail task: Terminus-MM labels up to 25

of the 30 seconds with the correct camera (partial score 0.83), but the task requires the camera of every second to be correct, so none of its runs counts as a pass. The Codex-CLI-only tasks illustrate why the main text does not treat native perception as a strict superset of terminal skill: their deliverables can be recovered through OCR, timestamped transcription, signal-energy analysis, synchronization repair, or deterministic file edits. The Terminus-MM-only tasks are enriched for per-event audio-visual records: the output often requires deciding, for each row, which spoken or sonic cue corresponds to which visual state, event, or action. The both-fail region contains the remaining 60 tasks; we list representative high-headroom cases rather than all 60 names because its main role in the paper is to characterize remaining benchmark headroom.

Table 21: **Tasks in each solver regime of Table 19.**

Regime	Tasks
Both solve (11)	`broadcast-package-edit`, `code-review-comment-attribution`, `interview-srt-refine`, `musical-mood-shot-pick`, `narration-drift-qc`, `near-duplicate-frame-dedup`, `proof-step-note`, `receipt-photo-to-json`, `semantic-image-retrieval`, `signal-based-qc-report`, `warehouse-sku-pack-audit`
Codex CLI only (6)	`av-desync-offset-repair`, `constant-hum-attenuation`, `debate-attribution`, `page-photo-to-text`, `stereo-channel-flip-repair`, `stream-alert-ack-audit`
Terminus-MM only (28)	`accessibility-sync-audit`, `adr-edit-detection`, `animation-narration-audit`, `birthday-money-shot`, `blind-audition-match`, `blood-test-pdfs-to-csv`, `bug-repro-claim-audit`, `constant-offset-srt`, `crm-compliance-audit`, `dead-air-removal`, `design-review-approval-audit`, `fugal-subject-entry-labeling`, `game-alert-mismatch`, `interview-music-ducking-audit`, `invoice-estimate-pdfs-to-xlsx`, `line-failure-annotation`, `lipsync-drift-correction`, `mock-call-automation`, `multi-mic-bleed-attribution`, `narration-mars-rover`, `prosody-take-selection`, `question-statement-intonation`, `screenshare-deictic-grounding`, `slack-action-extraction`, `speaker-action-attribution`, `spoken-decision-cell-ref`, `travel-clip-retrieval`, `tutorial-edit-recreation`
Both fail (60, representative)	`narration-visual-align`, `2-speaker-diarized-transcript-from-podcast-audio`, `av-privacy-exposure`, `multicam-active-speaker-cut`

Representative trajectory evidence. Table 22 gives one representative task from each regime. The examples are not used to define the regimes; they are used to interpret why the aggregate partition has the observed shape.

Synthesis. The regime analysis supports the interpretation in Section 4.3. Codex-CLI-only tasks show that strong terminal agents can still win when the media can be reduced to reliable derived signals (a transcript, an OCR dump, or a signal-feature table) and the main challenge is executing the right pipeline. Terminus-MM-only tasks show that native media grounding is valuable when the artifact depends on event-level correspondence between audio and visual evidence. The both-fail region shows that the next frontier is not simply stronger perception or stronger scripting in isolation, but agents that combine both: they must find the decisive media evidence and complete the terminal artifact reliably under a fixed interaction budget.

B.12 Filtering for the Cost Comparison in Section 4.2

This appendix supports the cost analysis in Section 4.2, which compares the cost of inspecting a missing modality through command-line tools to the cost of using native perception on the same task. Table 4 reports the seven Terminus rows.

Table 22: **Representative examples from the four solver regimes.**

Task	Codex CLI	Terminus-MM	Interpretation
`page-photo-to-text`	1.00	0.00	Codex CLI succeeds by constructing an OCR-centered document pipeline. The case illustrates that some media workflows are primarily pipeline-and-artifact problems rather than native-perception problems.
`accessibility-sync-audit`	0.00	1.00	The task requires aligning screen-reader audio with visual focus states and writing a row-level audit record. Codex CLI tries to inspect the audio-visual alignment by transcribing the audio and OCR'ing frames but does not finish in budget; Terminus-MM uses native A/V grounding to enumerate the events.
`interview-srt-refine`	0.955	0.960	Both systems reach near-perfect outputs. Their trajectories converge on similar transcript and boundary-refinement operations, so the task is solvable from a transcript alone.
`2-speaker-diarized-transcript-from-podcast-audio`	0.00	0.582	Both systems fall short of binary success. Codex CLI spends the run attempting to set up diarization pipelines; Terminus-MM identifies the speakers but lacks the required turn-boundary precision.

Table 23: **Matched-pair filter counts.** For each partial system: how many of the 105 tasks both it and Terminus-MM pass (Co-success), how many of those require a modality missing from the partial system (Modality required), and how many of those show the agent running command-line tools such as frame extraction, ASR, OCR, or signal-feature scripts to inspect the missing modality (Command-line attempts).

System	Missing access	Total	Co-success	Modality required	Command-line attempts
Terminus-2 (T)	image+audio+video	105	8	8	7
Terminus-KIRA (T+I)	audio+video	105	8	5	4
Terminus-A (T+A)	image+video	105	21	18	14
Terminus-V (T+V)	image+audio	105	25	25	13
Terminus-IA (T+I+A)	video	105	28	18	13
Terminus-IV (T+I+V)	audio	105	29	23	14
Terminus-AV (T+A+V)	image	105	24	7	4

Filtering procedure. For each partial system, a task enters the matched-pair set only if it passes three filters: both the partial system and Terminus-MM pass the task verifier; the task requires a modality missing from the partial system; and the partial trajectory shows the agent attempting to inspect the missing modality through command-line tools (extracted frames, transcripts, OCR dumps, or signal-feature tables). Table 23 shows the filter counts.

Where the overhead comes from. The high-overhead cases share a small set of trajectory patterns. Partial systems repeatedly try to extract the missing evidence by sampling frames or audio segments, running OCR or transcription, installing analysis libraries, computing signal features, and iteratively refining timestamps or candidate artifacts. These are valid terminal strategies, but they become costly when native perception would provide the decisive evidence directly.

B.13 Failure-Signature Definitions

Figure 5 assigns each binary-failed run of Terminus-MM and Codex CLI one primary failure signature from the following seven.

- *Timeout (tool setup)*: the run spends a large share of its turns or time installing tools and ends without a usable artifact.
- *Timeout (tool execution)*: the run keeps re-running a suitable processing pipeline with different parameters until the budget runs out, never producing a candidate answer.
- *Wrong (output format)*: values matching the gold answer appear in the agent's output near the end of the run, but its last tool calls do not write the required output file.
- *Wrong (wrong approach)*: the run commits to a processing pipeline that cannot yield the evidence the task needs and never produces a candidate answer.
- *Wrong (correct approach, low precision)*: the artifact identifies the right event or item but misses the verifier's tolerance in timing, overlap, or position.
- *Wrong (tool failure)*: the artifact rests on a conversion tool's output, such as cross-correlation, silence detection, or frame OCR, that does not carry the signal the task depends on.
- *Wrong (model reasoning)*: the artifact is well-formed and its timing or position is not the problem, but the decision itself is wrong, such as the wrong speaker or camera.