How GUI Agents Receive Instructions: An Exploratory Mapping of 85 Benchmark Families

Task Specification, Video Demonstrations and Evaluation Methods

Ankur Shrestha

Vertov, United States
ORCID: 0009-0009-6036-5849

14 November 2026

Abstract

Benchmarks for GUI and computer-use agents tell the agent what to do in different ways, and some now supply video or recorded demonstrations. We separate three constructs that descriptions of benchmarks often mix: how the goal is specified, what the agent observes while acting, and how completion is judged and what evidence can be inspected afterwards. We coded them for a declared, non-exhaustive frame of 101 candidate benchmarks: 61 whose abstracts mention video, demonstration or recording (stratum V) and a seeded random sample of 40 others (stratum C). Eighty-five benchmark families (147 components) could be read in full and are analyzed, each positive value tied to a verbatim quote, a location and a document hash (checked absences and not-applicable cases are recorded without a quote). Coding was done by AI agents, checked mechanically against the stored sources and audited by a second AI pass; no independent human recoding was performed, and the study was not preregistered. In both strata the goal is given as natural-language text in almost every component (80 of 81 in V, 64 of 66 in C). A video, recording or demonstration trajectory is part of the specification in 9 of 81 V components (8 families) and 1 of 66 C components, together with text in all but one, the primary condition of VideoGUI's high-level planning track, whose coding as a recording-only specification rests on one judgment about an outcome preview presented with a generic question wrapper. More V components are coded as using video as authoring input (18 of 81, some thinly evidenced) than as a specification channel. The contribution is a coding scheme, a quote-evidenced public registry and a reproducible pipeline for checking how benchmarks specify tasks; counts describe this sample, not the field.

Keywords: GUI agents; computer-use agents; benchmarks; task specification; video; demonstration; meta-research.

1 Introduction

A benchmark for a computer-use agent must tell the agent what to do. In most benchmarks this is a sentence. Some add an image, a recorded demonstration, an instructional video or a dialogue. Others show the agent a video of the screen while it acts, which is a different thing: the first is how the goal reaches the agent, the second is what the agent sees of its environment. A third construct is what the evaluator, and any later reader, can inspect to confirm that a task was completed. This paper keeps the three apart (Figure 1) and codes each for a declared sample of benchmarks.

Why the channel that carries the goal matters. A benchmark score reports how well an agent did given a particular way of being told. Two benchmarks that both say "tasks are given as natural-language instructions" can differ in whether an image, a demonstration or a dialogue is also needed to determine the goal, and a result on one cannot be read as a result on the other without

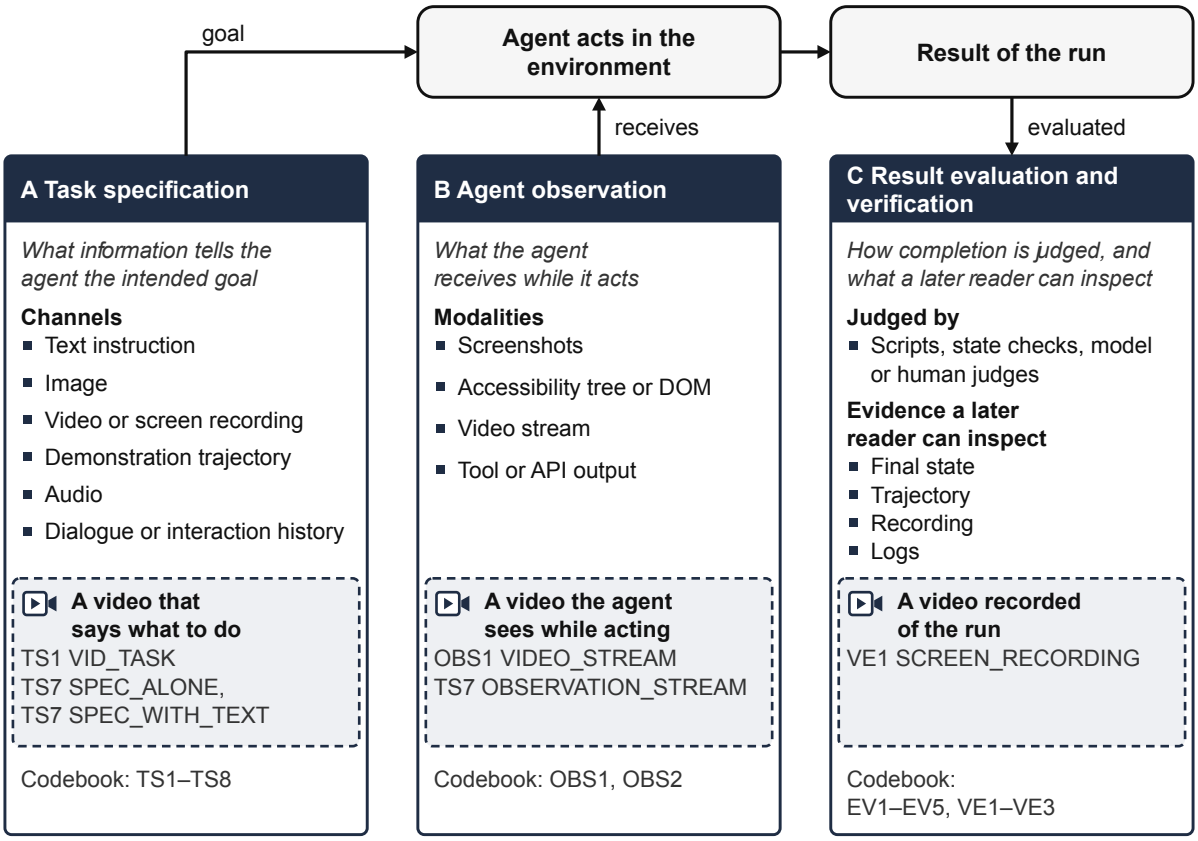


Figure 1: The three constructs coded in this study and where a video can sit.

knowing this. Benchmark authors and users need the information in order to compare benchmarks, to design new ones, and to say what an agent was and was not shown.

Why an instructional video is not an observation stream. A video that tells the agent what to do changes the task: the agent must infer a goal or a procedure from a recording. A video that the agent sees while acting changes only what it can perceive. A benchmark can have either, both or neither, and "video benchmark" in a survey or a leaderboard description often does not say which. Treating them as one would, for example, count a screen-recording observation stream as evidence that agents are instructed by demonstration.

Benchmarks that use video or demonstrations already exist, and several are discussed below. This paper does not claim that they are absent or rare. Its contribution is narrower. It records, benchmark by benchmark and component by component, how the task is specified, what the agent receives and what verification evidence exists, with a verbatim quote and a location for each value; it states the rules that separate the three constructs; and it releases the records and the pipeline so that others can check, reuse or dispute them. The study does not test whether showing an agent works better than telling it.

Research questions. RQ1: which channels carry the task goal (text, structured goal, image, video, demonstration, audio, dialogue)? RQ2: what role does a video or recording play in each benchmark, and in how many components is a recording part of the specification, or the only goal-bearing input in a named condition? RQ3: which observation modalities does the agent receive,

and how often is a video one of them? RQ4: how is success judged, and how often is the evaluator checked against human judgment? RQ5: how well supported are the codings? The questions follow a draft protocol that was not registered before coding (Section 3.7).

2 Related work

Surveys and taxonomies. Several surveys of GUI and computer-use agents carry benchmark tables. Nguyen et al. list about thirty benchmarks by domain, whether the entry is a dataset or an environment, and a free-text highlight [1]. Zhang et al. tabulate platform, year, whether the environment is live, data size and metric [2]. Hu et al. classify benchmarks by platform, static or interactive setting, simulated or real environment, and task type [3]; Tang et al. add evaluation method over more than forty datasets [4]. Sager et al. compare 33 datasets by observation and action space for agents driven by text instructions [5]. Yu et al. review 336 papers on software engineering for and with GUI agents and argue that a complete benchmark should state observation permissions, action space and judge type [6]. Surveys of agent evaluation organize benchmark design by data curation, environment, interface, metric and safety [7] or by evaluation objective and process [8]. In the tables and taxonomies we examined, the channel through which the task reaches the agent is not a column of its own; video appears at most as free text, for example the label "Video + Web" for one benchmark [1]. Three further surveys could not be read beyond their abstracts [9, 10, 11], and we make no statement about what their tables contain.

Instruction formats. Most interactive benchmarks give the task as a natural-language instruction. OSWorld frames its tasks as high-level instructions in real computer environments [12], and PixelHelp pairs English instructions with mobile UI action sequences [13]. Some benchmarks add images: in VisualWebArena, 25.2 percent of tasks include input images in the task objective [14]. Outside GUI work, VIMA interleaves text and images as a single task specification for robot manipulation [15]. The instruction format is therefore a design choice that can be stated per benchmark, which is what the coding scheme records.

Video- and demonstration-conditioned benchmarks. VideoGUI builds tasks from instructional videos and uses a visual query as its primary setting, with only a visual preview of the outcome given in that setting [16]. AssistGUI pairs a user query with an instructional video for reference [17]. WorldGUI, in its fifth version, describes each task as paired with a query, an instructional video and a project file [18]. VideoWebArena provides web tasks with video input, including manually crafted tutorial videos [19]. Ego2Web pairs egocentric video with web tasks [20]. OSWorld 2.0 includes a tutorial-following category [21]. VG-GUI-Bench asks agents to follow video tutorials [22]. LearnAct supplies demonstrations to a mobile agent [23], and UI-Vision is an offline desktop benchmark built from human-demonstration annotations [24]. OmniGUI takes video clips and audio as per-step inputs in smartphone episodes [25]. GUIDE studies understanding and assisting users from screen recordings [26]; a separate 2024 dataset with the same name addresses action prediction for process automation [27]. Secondary tabulations do not always agree with the primary papers: VideoGUI's comparison table lists AssistGUI under text queries [16], while the AssistGUI paper presents the query together with a video [17]. The two may use different definitions of the query, which is why this study records the role of a video from each benchmark's own text, with a quote and a location.

Programming by demonstration and instructional video. Showing a system what to do predates language-model agents; programming by demonstration is the subject of edited volumes from the 1990s and 2000s [28, 29]. SUGILITE lets a user demonstrate a task on a smartphone and generalizes a script from the demonstration, spoken instruction and the UI hierarchy [30]. Work

on instructional video links screencast tutorials to the running application [31] and recovers UI components from screen-captured video [32]; V2S translates recordings of app use into replayable scenarios [33]. These are methods, not benchmarks, and lie outside the frame.

Evaluation and verification. Benchmark quality has its own literature: BetterBench scores 24 benchmarks against 46 criteria [34], an agentic-benchmark checklist separates task validity, outcome validity and reporting [35], Raji et al. question the construct validity of benchmarks presented as general [36], and Kapoor et al. argue that agent evaluations should report cost with accuracy [37]. For GUI agents, automatic evaluators have been checked against human or oracle judgments [38, 39, 40], and an audit of failure-scored trajectories from five benchmarks found that a share of the failure verdicts were wrong [41]. On reproducibility, a study of 50 repositories proposes the rollout record as the unit of reproducibility [42] and a review of human baselines gives a reporting checklist [43]. A pilot audit of twelve benchmark papers by one auditor, with a released schema, is the closest methodological relative of this study [44].

How this study differs. It does not survey GUI agents, propose a taxonomy of benchmarks or evaluate any agent. It codes documents. For each component of each benchmark in a declared frame it records how the task is specified, what the agent receives, how success is judged and what evidence of completion exists, each value with a quote, a location, a read level and a verification status, and it keeps three constructs apart that tables in the surveys above do not separate. The frame is a keyword-targeted stratum plus a random comparison sample, not a census, and the coding is AI-assisted and has not been verified by a person.

3 Method

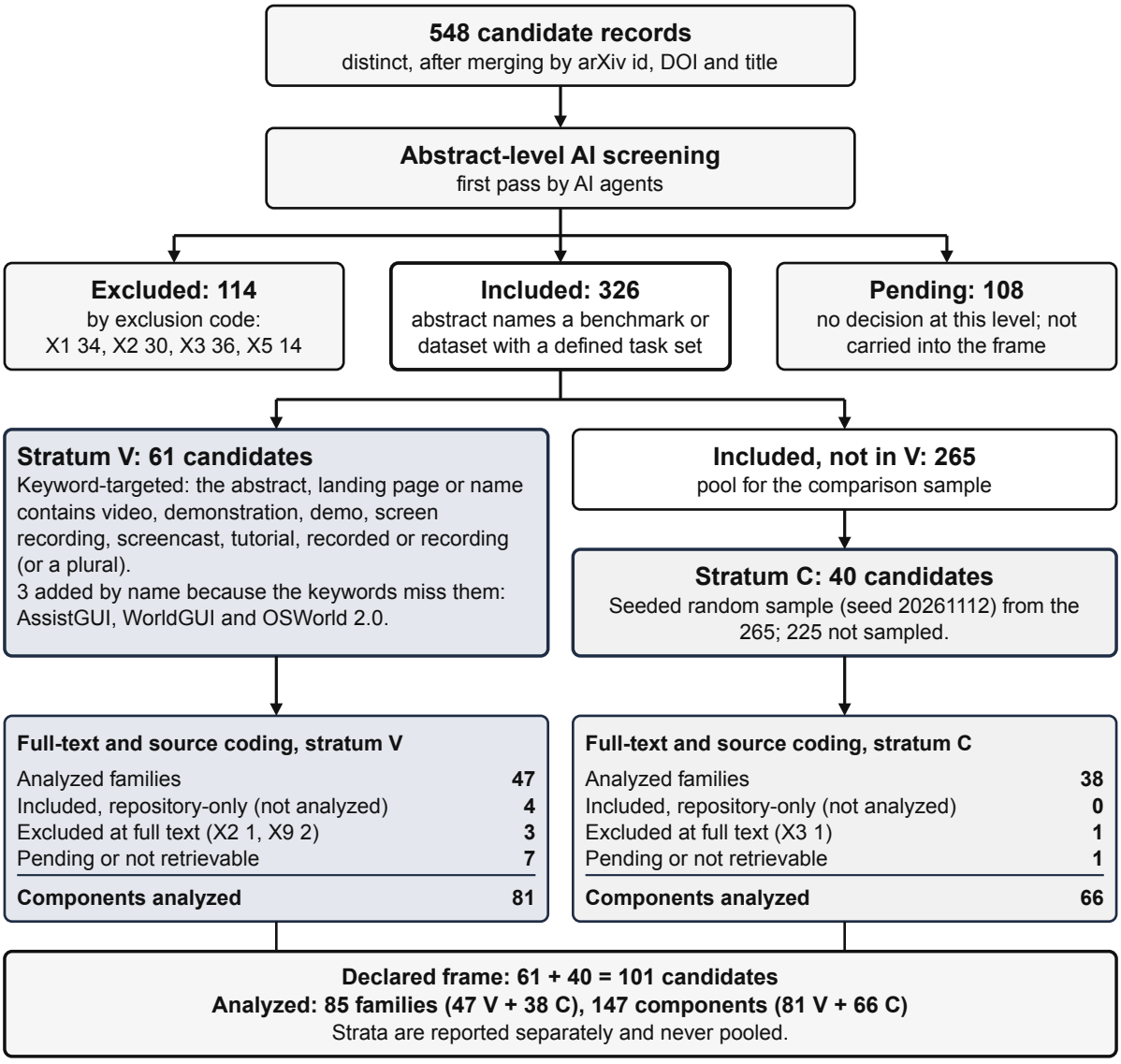
3.1 Study design

This is an exploratory mapping of a declared, non-exhaustive sample of benchmarks. It is not a systematic mapping study or a systematic review: the protocol was not registered before screening and coding, screening was done by AI agents, mostly on abstracts, without double screening, a reliability statistic or a recall check, and no independent human recoding was performed [45, 46]. The selection flow is laid out in the style of a PRISMA flow diagram for transparency (Figure 2); the study does not claim to follow the PRISMA statement [47].

3.2 Sampling and selection

The candidate universe came from two AI-run searches, one of 190 rows (184 distinct) and one of 460 rows, the second including 24 queries aimed at video-, demonstration- or recording-conditioned tasks, merged by arXiv id, DOI and title into 548 distinct candidates. AI agents screened them, mostly at abstract level, against written eligibility rules: 326 were included, 114 excluded and 108 left pending. The frame therefore carries the choices of those searches, which were steered toward video and so over-represent video-related benchmarks among the candidates, and no claim is made that it contains every relevant benchmark. Only included candidates enter the frame: 25 pending and 17 excluded candidates also matched the V keywords in their retrieved text and are outside the 85 analyzed families. No recall check was completed.

The declared frame, fixed on 2026-11-12 before the coding reported here, has two strata. Stratum V holds every included candidate whose retrieved abstract, landing page or name contains video, demonstration, demo, screen recording, screencast, tutorial, recorded, recording or screenshot sequence (or plurals), plus AssistGUI, WorldGUI and OSWorld 2.0 added by name because the keywords missed them: 61 candidates. Stratum C is a random sample of 40 (seed 20261112)



Not a census: the frame is a keyword-targeted stratum plus a seeded random comparison sample. Benchmarks outside it were not coded, and every screening and coding decision is an AI first pass, with no independent human recoding.

Exclusion codes: X1 method-only paper without a task set; X2 no agent action; X3 not GUI-based; X5 duplicate; X9 resource without an evaluation task set. Codes seen at abstract screening: X1, X2, X3, X5; at full text: X2, X3, X9.

Figure 2: Study selection. Candidate records, abstract-level screening, the declared frame (strata V and C) and the outcome of source-level coding. Counts are generated from the data files.

from the 265 included candidates outside V. The keyword list is deliberately broad, so V also catches benchmarks that merely mention human demonstrations as training data; the codebook, not the keyword, decides what counts as a video or demonstration role. The frame definition was chosen after the screening results and a pilot had been seen, which is a disclosed limitation. The coding agents' written brief used the working label "pre-registered census", which does not describe the study, and told them that Vertov's category terms (such as Show-First Work) are a strategy hypothesis to be neither used as search keys alone nor treated as findings; the brief is released with the data. The strata are never pooled.

A family is a named benchmark resource and its later versions; a derivative with its own task set is a separate family. A component is a track, split or task type inside a family whose task specification, observation space, evaluator or verification evidence differs. Family-level variables are coded once per family; other variables are coded for each component. Possible duplicate families under different names were flagged and not merged.

For each candidate the coder retrieved the paper, the official repository (README, license file, sample task files) and data cards where public, saved each file with its sha256, and recorded the document version and retrieval date. A family counted as analysable only when at least one primary document was read at full text for the task format, observation space, evaluation and artifact sections. Families whose primary document sat behind a browser challenge (OpenReview) or a paywall were retried through other open routes without bypassing any challenge; those still unreadable stayed pending.

What can and cannot be concluded: the counts describe the 85 analyzed families. They do not estimate how common any specification channel is among GUI-agent benchmarks, because V is defined by a keyword applied to abstracts and C is a small random sample of an AI-screened pool.

3.3 Constructs and codebook

Codebook 0.4.0-run1 has 31 variables. Task specification (TS1 to TS8) records which channels carry the goal, which single channel the goal cannot be determined without, whether text is withheld, and the role of any video or recording: specification alone, specification with text, optional context, observation stream, understanding target, authoring input (the authors used a video to build tasks and the agent does not receive it), content inside the environment, or none. Observation and context (OBS1, OBS2, CTX1, ENV1) records what the agent perceives while acting. Evaluation (EV1 to EV5) and verification evidence (VE1 to VE3) record how completion is judged and what a later reader can inspect. Identification and artifact variables (ID1 to ID7, AR1 to AR3) record versions, benchmark type, license and data availability. The role of a video is coded by what it does, never by its mere presence; a leaderboard score is not verification evidence. Every value has a status (directly reported, calculated, inferred, not reported, absent after checking, not applicable, unverified), a verbatim quote of at most 40 words with a location, the read level and a confidence rating. Variables with many not-reported or inferred values (for example TS6 ambiguity handling, not reported for 109 of 147 components) describe documentation, not design. Family and component boundaries follow the codebook rule and are judgment calls; possible duplicate families (for example SMAN-Bench and Mobile-Bench-v2) were not merged. The codebook was frozen for this round with its hash recorded; it was revised once after the pilot and once to settle open rules before coding.

3.4 Coding, quote check, audit and corrections

Thirteen AI coding agents coded 7 to 9 candidates each, blind to each other, plus two recovery coders for candidates whose primary document had not been retrievable. A schema validator checked all output. A script then checked that every quoted fragment of at least six words occurs verbatim in the stored source: of 4,687 coding rows, 2,794 carried a checkable quote, 2,793 passed and none failed (one was too short to test); 1,893 rows carry no quote because they record a checked absence or a not-applicable case. A passing quote shows that the sentence exists in the document, not that it supports the value.

To test support, four further AI auditors checked 251 rows drawn with a fixed seed: all 91 positive rows on TS1, TS7 and TS8, 60 of the 97 components coded as using no video (blind recoding), and 100 other rows (blind recoding). A final verification pass then re-checked, against

the stored sources, the rows that drive the headline findings: 203 checks in all (every video-related TS1 and TS7 positive row, the audit rows judged partly supported, a seeded sample of 25 text-goal components per stratum and every component without a text goal, and a seeded sample of evaluation rows). Of these, 178 were confirmed, 3 corrected, 21 left ambiguous and 1 could not be verified. Five values were changed in the registry after the audit and the verification pass (three TS7 values on one benchmark, MAGAR, whose episodes are frames and logs rather than video; one TS7 value on a critique benchmark; one evaluation value on a trust benchmark); each correction is in `corrections.csv` with the original value, the revised value and a reason, and the original is kept in the row's notes. The auditors were AI systems of the same family as the coders; the audit and verification are checks on extraction, not independent human recoding and not intercoder reliability. The 21 ambiguous cases were left as coded and are listed in the verification log; those that would move a headline number are examined in Sections 4.3 and 4.6.

3.5 Analysis

Descriptive only. For each coded value the number of components (and families with at least one such component) is reported with its denominator and a percentage, per stratum. No confidence intervals are reported: V is a keyword-defined set and C is a random sample of 40 from a pool of 265 AI-screened candidates, so an interval around a percentage would suggest an inference to the population of benchmarks that the design does not support. No hypothesis tests are run. Analysis code is standard-library Python, and the tables in this paper are generated from the result files, whose sha256 values are recorded with them.

3.6 Sensitivity analysis

Where a headline number depends on a disputed reading of one component, the count is re-derived under the alternative readings by a script (Table 5) and reported next to the main figure.

3.7 What this study is not

The study was not preregistered: the codebook, frame and analysis plan existed as an unregistered draft, coding took place before any registration, and the frame was defined after the screening and pilot results were seen, so all results are exploratory. Initial extraction, coding and auditing were AI-assisted and no independent human recoding was performed, so no reliability statistic between humans is reported; the agreement figures compare AI passes. A registration after the fact would document the protocol but would not make these results confirmatory.

4 Results

4.1 What was analyzed

Of 101 candidates, 85 families (47 in V, 38 in C; 147 components, 81 and 66) were read at full text and analyzed. Four more V candidates were coded from repositories only (SurveyAttentionBench, WebArena-Pro, RACE-Bench and PERCU) and are not analyzed. Four candidates were excluded after the document was read: GUIDE (X2, no acting agent; its task is understanding a recorded user), CLASHarness (X3, not GUI-based), and GroundCUA and "From the Wild Web to the Zoo" (X9, a resource without an evaluation task set). Eight candidates stayed pending because the primary document could not be read: seven in V and one in C, for example OpenReview-only

Table 1. Study selection: declared frame and analyzed sample

Stratum	Candidates in frame	Included, full text read (analytic)	Included, repository only (not analyzed)	Excluded at full text	Pending (primary document not retrievable)	Components analyzed
V	61	47	4	3 (X2=1; X9=2)	7	81
C	40	38	0	1 (X3=1)	1	66

papers that served a browser challenge. These losses are concentrated in recent unpublished work, so the analyzed sample leans toward papers with an arXiv version.

4.2 RQ1: channels that carry the goal

Table 2. Channels that carry the task goal to the agent (TS1, multi-valued)

Code	V: components k/n (%)	C: components k/n (%)	Families with the code (V / C)
TXT_NL	80/81 (99%)	64/66 (97%)	47 / 36
TXT_STRUCT	3/81 (4%)	1/66 (2%)	2 / 1
IMG_REF	8/81 (10%)	8/66 (12%)	5 / 5
VID_TASK	8/81 (10%)	1/66 (2%)	7 / 1
DEMO_TRAJ	1/81 (1%)	0/66 (0%)	1 / 0
DIALOG	4/81 (5%)	2/66 (3%)	4 / 2
AUDIO	2/81 (2%)	0/66 (0%)	2 / 0
OTHER	0/81 (0%)	1/66 (2%)	0 / 1
Components without a determinable value	0 of 81	0 of 66	

In both strata the goal is almost always carried by natural-language text: 80 of 81 components in V and 64 of 66 in C. Sixty of 81 V components (74 percent) and 54 of 66 C components (82 percent) are text-only; a non-text goal channel in addition to text occurs in 21 of 81 V components and 11 of 66 C components, most often a reference image (8 and 8 components). A video or recording is a goal channel in 8 of 81 V components (7 families) and 1 of 66 C components. One V component uses a demonstration trajectory as a goal channel (the demonstration-augmented condition of SCUBA), so a video or demonstration channel is present in 9 of 81 V components (8 families). For 71 of 81 V components and 61 of 66 C components the text channel alone is the one without which the goal cannot be determined (Table A1).

4.3 RQ2: role of video or recordings

SPEC_ALONE = the recording is the only goal-bearing input in at least one named condition; SPEC_WITH_TEXT = it illustrates the goal and text is also given; OBSERVATION_STREAM = the agent sees video while acting; CONTENT_IN_ENVIRONMENT = video is content the task acts on; AUTHORING_INPUT = used by the authors to build tasks but not given to the agent.

In V, 42 of 81 components (52 percent) use no video or recording in any role, against 59 of 66 in C (89 percent). In V the most common role is authoring input (18 components in 12 families): the authors built tasks from recorded demonstrations or videos, and the agent does not

Table 3. Role of any video or recording in the benchmark as released (TS7, multi-valued)

Code	V: components k/n (%)	C: components k/n (%)	Families with the code (V / C)
SPEC_ALONE	1/81 (1%)	0/66 (0%)	1 / 0
SPEC_WITH_TEXT	8/81 (10%)	1/66 (2%)	7 / 1
OPTIONAL_CONTEXT	3/81 (4%)	0/66 (0%)	3 / 0
OBSERVATION_STREAM	7/81 (9%)	0/66 (0%)	4 / 0
UNDERSTANDING_TARGET	4/81 (5%)	0/66 (0%)	3 / 0
AUTHORING_INPUT	18/81 (22%)	0/66 (0%)	12 / 0
CONTENT_IN_ENVIRONMENT	10/81 (12%)	6/66 (9%)	7 / 5
NONE	42/81 (52%)	59/66 (89%)	26 / 34
Components without a determinable value	0 of 81	0 of 66	

receive them. A video that is part of the specification is found in 8 V components (7 families: AssistGUI, Ego2Web, OSWorld 2.0, VG-GUI-Bench, VideoGUI, VideoWebArena, WorldGUI) and 1 C component (IndusGCC). Video that the agent sees only while acting (an observation stream) was coded in 7 V components, and video as the object of an understanding task in 4. The agent also sees a video stream among its observations in 8 of 80 V components with a determinable value, which is a different thing from a video that specifies the task (Section 4.4).

Table 4. Derived specification cells (components)

Cell	V: k/n (%)	C: k/n (%)
Text only (natural-language or structured text)	60/81 (74%)	54/66 (82%)
At least one non-text channel (image, video, demonstration, interaction history, audio or dialogue)	21/81 (26%)	11/66 (17%)
Neither text-only nor any listed non-text channel (other)	0/81 (0%)	1/66 (2%)
Video or demonstration trajectory is a goal channel (TS1)	9/81 (11%)	1/66 (2%)
Recording is the only goal input, in the benchmark's headline condition	1/81 (1%)	0/66 (0%)
Recording is the only goal input in at least one named condition	1/81 (1%)	0/66 (0%)
Video or demonstration channel, text withheld (TS3)	1/81 (1%)	0/66 (0%)
Video or demonstration channel, text also given (TS3)	8/81 (10%)	1/66 (2%)

The recording-only cell. One component, the high-level planning track of VideoGUI, is coded with a recording as the only goal-bearing input in a named condition, and that condition is the authors' primary one (1 of 81 V components; 0 of 66 C). We read the original paper, its task definition and the released data to settle the classification. The authors' "visual query" is a preview of the outcome, an image or a short clip of the finished effect, and the agent's task is to plan the sub-tasks; the benchmark's own text says that this primary setting provides "only a visual preview", that a textual condition and a combined condition are separate named conditions, and that the released data carry a full-task text field. It is an outcome preview given with a fixed, generic question wrapper and no task-specific text in the primary condition, not a recording of someone performing the steps. The codebook defines the recording-only code as the recording being the only goal-bearing input with no natural-language task text, and classes outcome previews and process demonstrations alike when no task text is given. Whether a generic question wrapper counts as natural-language task text is a judgment: our own verification pass marked this case ambiguous, and under a literal reading of the channel rule the wrapper is a text channel, the component is a recording with text, and the recording-only count is 0 of 81. The rule that any named condition counts was added after the pilot, in which VideoGUI was one of the video families, so it reflects this

benchmark. The cell therefore rests on one judgment about one component. Table 5 shows how the counts change under the alternative readings: under either alternative reading of VideoGUI the recording-only count in V is 0 of 81, not 1.

Table 5. Sensitivity of the video-in-specification counts to disputed codings (stratum V components; stratum C counts are unchanged by these readings)

Reading	Recording only (headline)	Video in specification (TS7)	Video or demo channel (TS1)
Current coding	1/81	8/81	9/81
VideoGUI planning read as visual plus text	0/81	8/81	9/81
VideoGUI planning read as an outcome preview, not a video channel	0/81	7/81	8/81
SCUBA demonstration condition not coded as a demonstration trajectory	1/81	8/81	8/81
WorldGUI optional video not coded as a specification channel	1/81	7/81	8/81
All three alternatives together	0/81	6/81	6/81

The other two disputed codings in Table 5 concern a demonstration trajectory in SCUBA's demonstration-augmented condition, which is a text plan summarizing a recorded human trajectory, and WorldGUI, whose video is released with every task but described by the coder as optional, while the version of the paper that we read describes each task as paired with a video. Under all three alternative readings together, a video or demonstration channel is part of the specification in 6 of 81 V components rather than 9.

Table 6. Components with a video, recording or demonstration channel in the specification, or a video role other than authoring or in-environment content (AI-coded; the full list of 147 components is in the registry)

Stratum	Benchmark (component)	TS1	TS7	TS8	Agent sees video (OBS1)
C	IndusGCC (main)	TXT_NL, VID_TASK	SPEC_WITH_TEXT	NO	yes
V	AgentStudio (idm-multiple)	TXT_NL	UNDERSTANDING_ TARGET	NO	no
V	AgentStudio (idm-single)	TXT_NL	UNDERSTANDING_ TARGET	NO	no
V	AgentStudio (online-compositional)	TXT_NL	OBSERVATION_ STREAM	NO	yes
V	AgentStudio (online-single-gui)	TXT_NL	OBSERVATION_ STREAM	NO	yes
V	AssistGUI (main)	TXT_NL, VID_TASK	SPEC_WITH_TEXT	NO	no
V	Ego2Web (main)	TXT_NL, VID_TASK	SPEC_WITH_TEXT	NO	no
V	GUI-Xplore (app-exploration-qa)	TXT_NL	UNDERSTANDING_ TARGET	NO	yes
V	GUI-Xplore (cross-app-automation)	TXT_NL	OPTIONAL_CONTEXT	NO	no
V	LivingScreen (l1-gui-action)	TXT_NL	CONTENT_IN_ ENVIRONMENT, OBSERVATION_ STREAM	NO	yes

Stratum	Benchmark (component)	TS1	TS7	TS8	Agent sees video (OBS1)
V	LivingScreen (l2-cross-source-understanding)	TXT_NL	CONTENT_IN_ ENVIRONMENT, OBSERVATION_ STREAM	NO	yes
V	LivingScreen (l3-closed-loop-application)	TXT_NL	CONTENT_IN_ ENVIRONMENT, OBSERVATION_ STREAM	NO	yes
V	OSWorld 2.0 (main)	DIALOG, TXT_NL, VID_TASK	AUTHORING_INPUT, CONTENT_IN_ ENVIRONMENT, SPEC_ WITH_TEXT	NO	no
V	OmniGUI (main)	AUDIO, TXT_NL	CONTENT_IN_ ENVIRONMENT, OBSERVATION_ STREAM	YES_ABLATION	yes
V	SCUBA (demonstration-augmented)	DEMO_ TRAJ, TXT_NL	NONE	NO	no
V	UGIF (ugif-dataset)	AUDIO, TXT_NL	NONE	NO	no
V	VG-GUI-Bench (video-guided-action-prediction)	TXT_NL, VID_TASK	SPEC_WITH_TEXT	NO	no
V	VideoCAD (cad-action-prediction)	IMG_REF	OBSERVATION_ STREAM	YES_ALL	yes
V	VideoCAD (cad-vqa)	TXT_NL	UNDERSTANDING_ TARGET	NO	no
V	VideoGUI (high-level-planning)	IMG_REF, TXT_NL, VID_TASK	AUTHORING_INPUT, SPEC_ALONE, SPEC_ WITH_TEXT	YES_HEADLINE	no
V	VideoWebArena (factual-retention)	TXT_NL, VID_TASK	SPEC_WITH_TEXT	NO	no
V	VideoWebArena (skill-retention)	TXT_NL, VID_TASK	OPTIONAL_CONTEXT, SPEC_WITH_TEXT	NO	no
V	WorldGUI (main)	TXT_NL, VID_TASK	OPTIONAL_CONTEXT, SPEC_WITH_TEXT	NO	no

4.4 RQ3: what the agent observes

Screenshots are provided in 73 of 80 V components with a determinable value and 58 of 66 C components; accessibility trees in 21 and 23; video streams in 8 of 80 V components and 1 of 66 C. Whether the observation space is prescribed by the benchmark or left to the agent method was recorded as a separate variable (OBS2): prescribed for 50 of 80 V and 37 of 66 C components. A video the agent sees while acting is therefore not the same set of benchmarks as those in which a recording is part of the task specification, which is why the two constructs are coded separately.

4.5 RQ4: evaluation

Success is judged by task-specific scripts, state queries, string or URL matches, element or action matches, geometric checks and language-model judges, often in combination within one component. Evaluator validation against human judgment is described for 28 of 80 V components with a

Table 7. Observation modalities provided to the agent (OBS1, multi-valued)

Code	V: components k/n (%)	C: components k/n (%)	Families with the code (V / C)
SCREENSHOT	73/80 (91%)	58/66 (88%)	44 / 34
A11Y_TREE	21/80 (26%)	23/66 (35%)	15 / 14
DOM_HTML	12/80 (15%)	8/66 (12%)	7 / 4
SET_OF_MARKS	6/80 (8%)	3/66 (5%)	3 / 3
OCR_TEXT	2/80 (2%)	2/66 (3%)	1 / 2
VIDEO_STREAM	8/80 (10%)	1/66 (2%)	5 / 1
TEXT_DESCRIPTION	1/80 (1%)	4/66 (6%)	1 / 3
TOOL_API_OUTPUT	8/80 (10%)	1/66 (2%)	4 / 1
SYSTEM_STATE	0/80 (0%)	1/66 (2%)	0 / 1
AUDIO	1/80 (1%)	0/66 (0%)	1 / 0
OTHER	0/80 (0%)	2/66 (3%)	0 / 2
NOT_REPORTED	2/80 (2%)	3/66 (5%)	2 / 2
Components without a determinable value	1 of 81	0 of 66	

Table 8. Success-evaluation methods (EV1, multi-valued)

Code	V: components k/n (%)	C: components k/n (%)	Families with the code (V / C)
EXEC_SCRIPT	23/81 (28%)	20/66 (30%)	15 / 14
STATE_QUERY	18/81 (22%)	19/66 (29%)	12 / 12
STRING_URL_DOM_MATCH	15/81 (19%)	19/66 (29%)	11 / 10
ELEMENT_ACTION_MATCH	23/81 (28%)	19/66 (29%)	16 / 11
GEOMETRIC	17/81 (21%)	9/66 (14%)	13 / 7
REFERENCE_TRAJECTORY_MATCH	9/81 (11%)	0/66 (0%)	6 / 0
LLM_JUDGE	16/81 (20%)	19/66 (29%)	14 / 12
HUMAN_JUDGE	4/81 (5%)	4/66 (6%)	4 / 4
REWARD_MODEL	1/81 (1%)	0/66 (0%)	1 / 0
OTHER	4/81 (5%)	2/66 (3%)	4 / 2
NOT_REPORTED	3/81 (4%)	2/66 (3%)	3 / 2
Components without a determinable value	0 of 81	0 of 66	

determinable value and 22 of 65 C components (Table A2), and a human baseline with a stated protocol for 10 and 6 components (Table A3). Task data are openly retrievable for 30 of 47 V families and 27 of 38 C families (Table A4). Two variables about verification evidence (VE1, run artifacts released; VE2, what the verifier reads) could not be coded consistently, as Appendix B explains; their tables are in the appendix and are not interpreted.

4.6 RQ5: quality of the evidence

In the AI audit, 77 of 91 positive rows were fully supported, 12 partly, 2 had the wrong value and none was unsupported (Table A8). Both wrong values (on TS7) were corrected before the counts in this paper were computed. In blind recoding of 160 rows (Table A9) the auditor matched the coder's value set exactly in 141 (88 percent): in all 17 ID7 rows, all 11 TS1 rows and all 60 rows coded as using no video in any role, and weakly for VE1 and VE2 (3 of 8 and 6 of 13). Of the 97 components coded as using no video in any role when the audit was drawn, 60 were rechecked blind and all 60 were confirmed. Where the pilot and this registry both coded a family, TS1 matched in

10 of 11 and TS7 in 7 of 11; the TS7 differences follow from a rule change made between the two rounds. The 172 rows of documented disagreements between a paper and its repository, or between parts of one paper (for example task counts), cover 73 families; they are recorded as reported, have not been verified by a person, and are not errors until confirmed.

The 21 ambiguous cases from the verification pass are kept as coded. Reading some of them the other way would change the counts above: VideoGUI (recording-only 1 of 81 becomes 0), SCUBA (video or demonstration channel 9 of 81 becomes 8), AgentStudio's identification tasks (understanding target 4 becomes 2), World of Bits and OSWorld 2.0 (authoring input 18 becomes 15 if all three components are read the other way), AgentStudio's online tasks and VideoCAD (observation stream 7 becomes as few as 4), META-GUI, whose goal is carried by replayed dialogue and could also count as text (text goal in C 64 becomes 65 of 66), and the image channels of MLA-Trust privacy and of one puzzle benchmark (a non-text goal channel in C, 11 becomes 9 of 66). The verification log lists each case with its quote.

5 Discussion

In the coded sample, task goals in GUI and computer-use agent benchmarks are overwhelmingly given as text, in both the video-targeted and the comparison stratum. Video does appear. In the targeted stratum a recording or demonstration is part of the specification in about one component in nine (9 of 81), together with text except for the VideoGUI condition, and in 1 of 66 comparison components. Reading the video-related benchmarks by role shows why the distinction between specification and observation matters: of the components with a video role in V, more were coded as authoring input (18) than as a video channel of the specification (8), and the agent sees video while acting in 8 of 80. A benchmark described as a "video" benchmark may therefore never give the agent a recording to follow.

The counts do not say how common video is among GUI benchmarks: V was built from a keyword that over-includes, C is a small random sample of what remained, and 12 of 101 candidates could not be fully analyzed. They also do not say that video-instructed agents do better or worse than text-instructed agents; the study did not test that.

What the registry enables. Each benchmark has a quoted, located, hashed record of how it specifies the task, so a statement such as "benchmark X gives the agent a video and a text query" can be checked against the source. Researchers can use it to compare task-input modalities across benchmark families, to find benchmarks whose evaluator reads a final state, a trajectory or a recording, to design benchmarks that fill specification channels that are rare in the sample, and to dispute a coding with a better quote. Maintainers receive a list of places where their paper and release disagree, for them to confirm or correct.

What limits broader conclusions: the sample is declared and not exhaustive, coding was AI-assisted and not independently recoded by people, the audit used models from the coders' family, the study was not preregistered, and several definitions (what counts as a video, a demonstration or a visual preview) are judgment calls, as the VideoGUI case shows. Settling them would need people reading the sources, and the disputed rows are marked for that purpose.

6 Limitations

Sampling. The frame is non-exhaustive and partly selected by keyword, so the results describe this sample and should not be generalized to all GUI-agent benchmarks. The comparison stratum

is a small random sample, and some sources could not be retrieved or were read only through their repositories (Section 4.1).

Classification. Coding involved judgment. Family and component boundaries, and the line between a specification video, an observation stream and an authoring input, depend on the codebook's definitions, and a small number of borderline classifications are sensitive to them. The sensitivity analyses (Section 3.6, Table 5) and the evidence records in the registry allow readers to examine and contest individual codings.

Verification. The manuscript was reviewed by the author and additional readers. Data extraction and classification were AI-assisted, with the methodology and validation procedures detailed in Section 3.

Scope. Benchmarks change between versions; each record names the version read and the retrieval date. Two evaluation-evidence variables (VE1, VE2) were insufficiently consistent to interpret (Appendix B). The study was not preregistered, so all results are exploratory.

Statements

Data and materials availability. The registry (families, components and an evidence register with a verbatim quote, location, source URL, document hash and verification status for every value), the codebook and coding manual, the data dictionary, the correction log, the license assessment and the analysis scripts are released on Zenodo (dataset DOI: https://doi.org/10.5281/zenodo.24217767; data CC BY 4.0, code MIT). Source documents are third-party works and are not redistributed; hashes and URLs allow verification.

Competing interests. The author is affiliated with Vertov (https://vertov.app), a company developing screen-recording and AI-agent collaboration software. Vertov has a commercial interest in the subject of this research.

Funding. This research was funded by Vertov. No external funding was received.

Use of AI. Claude (Anthropic) was used to assist with literature research, data extraction and classification, analysis, verification checks and manuscript preparation. The methods and limitations of AI-assisted coding are described in Sections 3 and 6.

Ethics. The study analyzed published benchmark papers, repositories and data cards. No human participants and no personal data were involved.

Author contributions. Ankur Shrestha: Conceptualization, methodology, investigation, analysis, writing and project supervision.

References

[1] D. Nguyen, J. Chen, Y. Wang, G. Wu, N. Park, Z. Hu, H. Lyu, J. Wu, R. Aponte, Y. Xia, X. Li, J. Shi, H. Chen, V. D. Lai, Z. Xie, S. Kim, R. Zhang, T. Yu, M. Tanjim, N. K. Ahmed, P. Mathur, S. Yoon, L. Yao, B. Kveton, J. Kil, T. H. Nguyen, T. Bui, T. Zhou, R. A. Rossi, and F. Dernoncourt, "GUI Agents: A Survey," in *Findings of the Association for Computational Linguistics: ACL 2025*, 2025, pp. 22 522–22 538. [Online]. Available: https://doi.org/10.18653/v1/2025.findings-acl.1158

[2] C. Zhang, S. He, J. Qian, B. Li, L. Li, S. Qin, Y. Kang, M. Ma, G. Liu, Q. Lin, S. Rajmohan, D. Zhang, and Q. Zhang, "Large Language Model-Brained GUI Agents: A Survey," arXiv preprint, v12, 2024. [Online]. Available: https://arxiv.org/abs/2411.18279

[3] X. Hu, T. Xiong, B. Yi, Z. Wei, R. Xiao, Y. Chen, J. Ye, M. Tao, X. Zhou, Z. Zhao, Y. Li, S. Xu, S. Wang, X. Xu, S. Qiao, Z. Wang, K. Kuang, T. Zeng, L. Wang, J. Li, Y. E. Jiang, W. Zhou, G. Wang, K. Yin, Z. Zhao, H. Yang, F. Wu, S. Zhang, and F. Wu, "OS Agents: A Survey on MLLM-based Agents for Computer, Phone and Browser Use," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 7436–7465. [Online]. Available: https://doi.org/10.18653/v1/2025.acl-long.369

[4] F. Tang, H. Xu, H. Zhang, S. Chen, X. Wu, Y. Shen, W. Zhang, G. Hou, Z. Tan, Y. Yan, K. Song, J. Shao, W. Lu, J. Xiao, and Y. Zhuang, "A Survey on (M)LLM-Based GUI Agents," arXiv preprint, v2, 2025. [Online]. Available: https://arxiv.org/abs/2504.13865

[5] P. J. Sager, B. Meyer, P. Yan, R. von Wartburg-Kottler, L. Etaiwi, A. Enayati, G. Nobel, A. Abdulkadir, B. F. Grewe, and T. Stadelmann, "A Comprehensive Survey of Agents for Computer Use: Foundations, Challenges, and Future Directions," *Journal of Artificial Intelligence Research*, vol. 85, 2026. [Online]. Available: https://doi.org/10.1613/jair.1.19490

[6] S. Yu, Y. Ling, J. Xing, Q. Zhou, C. Fang, and Z. Chen, "Software Engineering for and with GUI Agent," arXiv preprint, 2026. [Online]. Available: https://arxiv.org/abs/2608.09278

[7] A. Yehudai, L. Eden, A. Li, G. Uziel, Y. Zhao, R. Bar-Haim, A. Cohan, and M. Shmueli-Scheuer, "Survey on Evaluation of LLM-based Agents," arXiv preprint, v2, 2025. [Online]. Available: https://arxiv.org/abs/2503.16416

[8] M. Mohammadi, Y. Li, J. Lo, and W. Yip, "Evaluation and Benchmarking of LLM Agents: A Survey," in *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, 2025, pp. 6129–6139. [Online]. Available: https://doi.org/10.1145/3711896.3736570

[9] Y. Chen, S. Wang, Y. Wang, and Q. Yao, "A Survey on Benchmarks of LLM-based GUI Agents," Transactions on Machine Learning Research (June 2026); preprint on TechRxiv, posted 2025-12-16, 2026. [Online]. Available: https://doi.org/10.36227/techrxiv.176591818.87526814/v1

[10] L. Huang, J. Liu, C. Wang, R. Xu, W. Xu, Z. Xu, Q. Zhang, Y. Zhang, K. Fu, L. Gao, Y. Xu, L. Zhang, L. Guo, and S. Xu, "GA: A Comprehensive Survey on LLM-based GUI Agent," TechRxiv preprint, posted 2025-07-22, 2025. [Online]. Available: https://doi.org/10.36227/techrxiv.175322671.12208812/v1

[11] L. Zu, K. Huang, G. Xu, J. Fan, B. Wu, S. He, H. Fan, X. Xu, C. Li, Y. Li, S. Ying, J. Yan, B. Jing, M. Li, and Z. Zhu, "GUI Grounding for Computer-Use Agents: A Survey," Preprints.org, posted 2026-11-04, 2026. [Online]. Available: https://doi.org/10.20944/preprints202611.0431.v1

[12] T. Xie, D. Zhang, J. Chen, X. Li, S. Zhao, R. Cao, T. J. Hua, Z. Cheng, D. Shin, F. Lei, Y. Liu, Y. Xu, S. Zhou, S. Savarese, C. Xiong, V. Zhong, and T. Yu, "OSWorld: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments,"

in *Advances in Neural Information Processing Systems 37*, 2024, pp. 52 040–52 094. [Online]. Available: https://doi.org/10.52202/079017-1650

[13] Y. Li, J. He, X. Zhou, Y. Zhang, and J. Baldridge, "Mapping Natural Language Instructions to Mobile UI Action Sequences," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 8198–8210. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.729

[14] J. Y. Koh, R. Lo, L. Jang, V. Duvvur, M. Lim, P.-Y. Huang, G. Neubig, S. Zhou, R. Salakhutdinov, and D. Fried, "VisualWebArena: Evaluating Multimodal Agents on Realistic Visual Web Tasks," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024, pp. 881–905. [Online]. Available: https://doi.org/10.18653/v1/2024.acl-long.50

[15] Y. Jiang, A. Gupta, Z. Zhang, G. Wang, Y. Dou, Y. Chen, L. Fei-Fei, A. Anandkumar, Y. Zhu, and L. Fan, "VIMA: Robot Manipulation with Multimodal Prompts," in *Proceedings of the 40th International Conference on Machine Learning*, 2023, pp. 14 975–15 022. [Online]. Available: https://proceedings.mlr.press/v202/jiang23b.html

[16] K. Q. Lin, L. Li, D. Gao, Q. WU, M. Yan, Z. Yang, L. Wang, and M. Z. Shou, "VideoGUI: A Benchmark for GUI Automation from Instructional Videos," in *Advances in Neural Information Processing Systems 37*, 2024, pp. 69 329–69 360. [Online]. Available: https://doi.org/10.52202/079017-2214

[17] D. Gao, L. Ji, Z. Bai, M. Ouyang, P. Li, D. Mao, Q. Wu, W. Zhang, P. Wang, X. Guo, H. Wang, L. Zhou, and M. Z. Shou, "ASSISTGUI: Task-Oriented Desktop Graphical User Interface Automation," arXiv preprint, v2, 2023. [Online]. Available: https://arxiv.org/abs/2312.13108

[18] H. H. Zhao, K. Yang, W. Yu, D. Gao, and M. Z. Shou, "WorldGUI: An Interactive Benchmark for Desktop GUI Automation from Any Starting Point," arXiv preprint v5, 2025. [Online]. Available: https://arxiv.org/abs/2502.08047

[19] L. Jang, Y. Li, D. Zhao, C. Ding, J. Lin, P. P. Liang, R. Bonatti, and K. Koishida, "VideoWebArena: Evaluating Long Context Multimodal Agents with Video Understanding Web Tasks," in *International Conference on Learning Representations (ICLR 2025)*, 2025. [Online]. Available: https://arxiv.org/abs/2410.19100

[20] S. Yu, L. Shu, A. Yang, Y. Fu, S. Sunkara, M. Wang, J. Chen, M. Bansal, and B. Gong, "Ego2Web: A Web Agent Benchmark Grounded in Egocentric Videos," arXiv preprint, 2026. [Online]. Available: https://arxiv.org/abs/2603.22529

[21] M. Yuan, Z. Zhou, X. Xiong, W. Wu, J. Sun, J. Song, K. Cui, B. Wang, H. Wu, Y. Li, D. Lu, H. Lu, Q. Zhen, X. Wang, J. Deng, Y. Yang, C. Chen, B. Zheng, A. Su, X. Yu, H. Zou, S. Agashe, X. H. Lu, M. Kaur, Z. Qi, V. S. Chen, F. Sala, D. Liu, J. Lin, Z. Yu, Y. Su, S. Reddy, X. E. Wang, P. Qi, T. Xie, and T. Yu, "OSWorld 2.0: Benchmarking Computer Use Agents on Long-Horizon Real-World Tasks," arXiv preprint, v2, 2026. [Online]. Available: https://arxiv.org/abs/2606.29537

[22] S. Fan, Q. Liu, R. Yin, M.-H. Guo, and S. Yang, "Bridging VideoQA and Video-Guided Agentic Tasks via Generalized Keyframe Extraction," arXiv preprint, 2026. [Online]. Available: https://arxiv.org/abs/2606.29445

[23] G. Liu, P. Zhao, L. Liu, Z. Chen, Y. Chai, S. Ren, H. Wang, S. He, and W. Meng, "LearnAct: Few-Shot Mobile GUI Agent with a Unified Demonstration Benchmark," arXiv preprint, 2025. [Online]. Available: https://arxiv.org/abs/2504.13805

[24] S. Nayak, X. Jian, K. Q. Lin, J. A. Rodriguez, M. Kalsi, N. Chapados, M. T. Özsu, A. Agrawal, D. Vazquez, C. Pal, P. Taslakian, S. Gella, and S. Rajeswar, "UI-Vision: A Desktop-centric GUI Benchmark for Visual Perception and Interaction," in *Proceedings of the 42nd International Conference on Machine Learning*, 2025, pp. 45 817–45 851. [Online]. Available: https://proceedings.mlr.press/v267/nayak25a.html

[25] F. Henry, L. Lin, J. Zhu, Yangfan, B. Zhang, M. Chen, and S. Huang, "OmniGUI: Benchmarking GUI Agents in Omni-Modal Smartphone Environments," arXiv preprint, v2, 2026. [Online]. Available: https://arxiv.org/abs/2605.18758

[26] S. Yang, J. Yu, Y.-H. Peng, K. Q. Lin, J. W. Cho, Y. Song, and J. Kim, "GUIDE: A Benchmark for Understanding and Assisting Users in Open-Ended GUI Tasks," arXiv preprint, 2026. [Online]. Available: https://arxiv.org/abs/2603.25864

[27] R. Chawla, A. Jha, M. Kumar, M. NS, and I. Bhola, "GUIDE: Graphical User Interface Data for Execution," arXiv preprint, v2, 2024. [Online]. Available: https://arxiv.org/abs/2404.16048

[28] A. Cypher *et al.*, Eds., *Watch What I Do: Programming by Demonstration*. MIT Press, 1993. [Online]. Available: https://lccn.loc.gov/93018319

[29] H. Lieberman, Ed., *Your Wish Is My Command: Programming by Example*. Morgan Kaufmann, 2001. [Online]. Available: https://lccn.loc.gov/00069638

[30] T. J.-J. Li, A. Azaria, and B. A. Myers, "SUGILITE: Creating Multimodal Smartphone Automation by Demonstration," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 2017, pp. 6038–6049. [Online]. Available: https://doi.org/10.1145/3025453.3025483

[31] S. Pongnumkul, M. Dontcheva, W. Li, J. Wang, L. Bourdev, S. Avidan, and M. F. Cohen, "Pause-and-play: automatically linking screencast video tutorials with applications," in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, 2011, pp. 135–144. [Online]. Available: https://doi.org/10.1145/2047196.2047213

[32] N. Banovic, T. Grossman, J. Matejka, and G. Fitzmaurice, "Waken: reverse engineering usage information and interface structure from software videos," in *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, 2012, pp. 83–92. [Online]. Available: https://doi.org/10.1145/2380116.2380129

[33] C. Bernal-Cárdenas, N. Cooper, K. Moran, O. Chaparro, A. Marcus, and D. Poshyvanyk, "Translating Video Recordings of Mobile App Usages into Replayable Scenarios," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, 2020, pp. 309–321. [Online]. Available: https://doi.org/10.1145/3377811.3380328

[34] A. Reuel, A. Hardy, C. Smith, M. Lamparth, M. Hardy, and M. J. Kochenderfer, "BetterBench: Assessing AI Benchmarks, Uncovering Issues, and Establishing Best Practices," in *Advances in Neural Information Processing Systems 37*, 2024, pp. 21 763–21 813. [Online]. Available: https://doi.org/10.52202/079017-0685

[35] Y. Zhu, T. Jin, Y. Pruksachatkun, A. Zhang, S. Liu, S. Cui, S. Kapoor, S. Longpre, K. Meng, R. Weiss, F. Barez, R. Gupta, J. Dhamala, J. Merizian, M. Giulianelli, H. Coppock, C. Ududec, A. Kellermann, J. Sekhon, J. Steinhardt, S. Schwettmann, A. Narayanan, M. A. Zaharia, I. Stoica, P. Liang, and D. Kang, "Establishing Best Practices in Building Rigorous Agentic Benchmarks," in *Advances in Neural Information Processing Systems 38*, 2025, pp. 184 435–184 475. [Online]. Available: https://doi.org/10.52202/085713-5547

[36] I. D. Raji, E. M. Bender, A. Paullada, E. Denton, and A. Hanna, "AI and the Everything in the Whole Wide World Benchmark," in *NeurIPS 2021 Datasets and Benchmarks Track*, 2021. [Online]. Available: https://arxiv.org/abs/2111.15366

[37] S. Kapoor, B. Stroebl, Z. S. Siegel, N. Nadgir, and A. Narayanan, "AI Agents That Matter," arXiv preprint, 2024. [Online]. Available: https://arxiv.org/abs/2407.01502

[38] T. Xue, W. Qi, T. Shi, C. H. Song, B. Gou, D. Song, H. Sun, and Y. Su, "An Illusion of Progress? Assessing the Current State of Web Agents," in *Conference on Language Modeling (COLM 2025)*, 2025. [Online]. Available: https://arxiv.org/abs/2504.01382

[39] J. Pan, Y. Zhang, N. Tomlin, Y. Zhou, S. Levine, and A. Suhr, "Autonomous Evaluation and Refinement of Digital Agents," in *Conference on Language Modeling (COLM 2024)*, 2024. [Online]. Available: https://arxiv.org/abs/2404.06474

[40] X. H. Lù, A. Kazemnejad, N. Meade, A. Patel, D. Shin, A. Zambrano, K. Stańczak, P. Shaw, C. J. Pal, and S. Reddy, "AgentRewardBench: Evaluating Automatic Evaluations of Web Agent Trajectories," arXiv preprint, v2, 2025. [Online]. Available: https://arxiv.org/abs/2504.08942

[41] Z. Dong, Z. Ma, Z. Wang, Y. Li, Z. Liu, R. Deng, Q. Zhan, and R. Qian, "How Benchmarks Mis-Score Computer-Use Agents," arXiv preprint, 2026. [Online]. Available: https://arxiv.org/abs/2607.28367

[42] C. Masters, Z. Liu, and S. V. Albrecht, "Rollout Cards: A Reproducibility Standard for Agent Research," arXiv preprint, 2026. [Online]. Available: https://arxiv.org/abs/2605.12131

[43] K. L. Wei, P. Paskov, S. Dev, M. J. Byun, A. Reuel, X. Roberts-Gaal, R. Calcott, E. Coxon, and C. Deshpande, "Recommendations and Reporting Checklist for Rigorous & Transparent Human Baselines in Model Evaluations," in *Proceedings of the 42nd International Conference on Machine Learning*, 2025, pp. 82 265–82 325. [Online]. Available: https://arxiv.org/abs/2506.13776

[44] M. N. Moghadasi and F. Ghaderi, "What Twelve LLM Agent Benchmark Papers Disclose About Themselves: A Pilot Audit and an Open Scoring Schema," arXiv preprint, 2026. [Online]. Available: https://arxiv.org/abs/2605.21404

[45] B. Kitchenham and S. Charters, "Guidelines for performing Systematic Literature Reviews in Software Engineering," Keele University and University of Durham, Tech. Rep. EBSE-2007-01, 2007. [Online]. Available: https://gsort.ifba.edu.br/dl175

[46] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic Mapping Studies in Software Engineering," in *12th International Conference on Evaluation and Assessment in Software Engineering (EASE 2008)*, 2008. [Online]. Available: https://doi.org/10.14236/ewic/ease2008.8

[47] M. J. Page, J. E. McKenzie, P. M. Bossuyt, I. Boutron, T. C. Hoffmann, C. D. Mulrow, L. Shamseer, J. M. Tetzlaff, E. A. Akl, S. E. Brennan, R. Chou, J. Glanville, J. M. Grimshaw, A. Hróbjartsson, M. M. Lalu, T. Li, E. W. Loder, E. Mayo-Wilson, S. McDonald, L. A. McGuinness, L. A. Stewart, J. Thomas, A. C. Tricco, V. A. Welch, P. Whiting, and D. Moher, "The PRISMA 2020 statement: an updated guideline for reporting systematic reviews," *BMJ*, vol. 372, p. n71, 2021. [Online]. Available: https://doi.org/10.1136/bmj.n71

A Additional tables

Table A1. Single channel without which the goal cannot be determined (TS2)

Code	V: components k/n (%)	C: components k/n (%)	Families with the code (V / C)
TXT_NL	71/81 (88%)	61/66 (92%)	44 / 36
IMG_REF	2/81 (2%)	0/66 (0%)	2 / 0
DIALOG	1/81 (1%)	1/66 (2%)	1 / 1
MIXED_JOINT	6/81 (7%)	2/66 (3%)	4 / 2
VARIES_BY_TASK	1/81 (1%)	1/66 (2%)	1 / 1
OTHER	0/81 (0%)	1/66 (2%)	0 / 1
Components without a determinable value	0 of 81	0 of 66	

Table A2. Evaluator validated against human judgment (EV4)

Code	V: components k/n (%)	C: components k/n (%)	Families with the code (V / C)
YES_QUANTIFIED	13/80 (16%)	14/65 (22%)	11 / 8
YES_QUALITATIVE	15/80 (19%)	8/65 (12%)	8 / 5
NO	41/80 (51%)	35/65 (54%)	23 / 22
NOT_REPORTED	11/80 (14%)	8/65 (12%)	7 / 4
Components without a determinable value	1 of 81	1 of 66	

Table A3. Human baseline (EV5)

Code	V: components k/n (%)	C: components k/n (%)	Families with the code (V / C)
REPORTED_WITH_PROTOCOL	10/80 (12%)	6/66 (9%)	9 / 6
REPORTED_WITHOUT_PROTOCOL	13/80 (16%)	5/66 (8%)	6 / 5
NONE	57/80 (71%)	52/66 (79%)	33 / 29
NOT_REPORTED	0/80 (0%)	3/66 (5%)	0 / 1
Components without a determinable value	1 of 81	0 of 66	

Table A4. Task data openly retrievable (AR3; one value per family)

Code	V: families k/n (%)	C: families k/n (%)	Families with the code (V / C)
OPEN	30/47 (64%)	27/38 (71%)	30 / 27
GATED	1/47 (2%)	0/38 (0%)	1 / 0
PARTIAL	10/47 (21%)	6/38 (16%)	10 / 6
CLOSED	5/47 (11%)	4/38 (11%)	5 / 4
LINK_BROKEN	1/47 (2%)	1/38 (3%)	1 / 1
Families without a determinable value	0 of 47	0 of 38	

Table A5. Benchmark type (ID7)

Code	V: components k/n (%)	C: components k/n (%)	Families with the code (V / C)
TASK_ONLINE	43/81 (53%)	38/66 (58%)	28 / 24
TASK_OFFLINE	23/81 (28%)	9/66 (14%)	17 / 8
GROUNDING	6/81 (7%)	4/66 (6%)	5 / 3
SAFETY_ROBUSTNESS	3/81 (4%)	15/66 (23%)	2 / 9
SCREEN_UNDERSTANDING	4/81 (5%)	0/66 (0%)	3 / 0
OTHER	2/81 (2%)	0/66 (0%)	2 / 0
Components without a determinable value	0 of 81	0 of 66	

B Verification-evidence variables (VE1, VE2): exploratory and not interpretable

VE1 (run artifacts released or retained) and VE2 (what the verifier reads) were coded in the same way as the other variables, but in the blind AI recoding only 3 of 8 and 6 of 13 audited rows matched the original value. The weakness is in the variables, not only in the coders. VE1 depends on whether a benchmark releases trajectories, recordings or logs, which papers describe loosely and repositories change after publication; VE2 is multi-valued and often has to be inferred from evaluator code that the coders did not run. We therefore show the tables for transparency, treat them as exploratory, and draw no conclusion from them. A stronger design would define each code from released evaluator code and files rather than from prose, require two independent coders with adjudication, and be tested for agreement on a pilot before a full round.

Table A6. What the verifier reads (VE2, multi-valued). Exploratory; unreliable in the AI audit

Code	V: components k/n (%)	C: components k/n (%)	Families with the code (V / C)
FINAL_STATE	27/81 (33%)	24/66 (36%)	18 / 16
DOM_URL	3/81 (4%)	8/66 (12%)	3 / 3
SCREENSHOTS	6/81 (7%)	3/66 (5%)	6 / 3
ACTION_TRACE	31/81 (38%)	25/66 (38%)	23 / 15
AGENT_SELF_REPORT	2/81 (2%)	3/66 (5%)	1 / 1
HUMAN_REVIEW_OF_RUN	2/81 (2%)	3/66 (5%)	2 / 3
OTHER	23/81 (28%)	19/66 (29%)	16 / 12
NOT_REPORTED	5/81 (6%)	4/66 (6%)	5 / 4
Components without a determinable value	0 of 81	0 of 66	

VE1 and VE2 had the lowest agreement in the AI audit (Table A9). Shown for transparency only; do not interpret the values.

Table A7. Run artifacts released or retained (VE1). Exploratory; unreliable in the AI audit

Code	V: components k/n (%)	C: components k/n (%)	Families with the code (V / C)
ACTION_TRAJECTORY	9/48 (19%)	7/51 (14%)	8 / 5
PER_STEP_SCREENSHOTS	6/48 (12%)	5/51 (10%)	5 / 3
SCREEN_RECORDING	3/48 (6%)	8/51 (16%)	3 / 4
LOGS	3/48 (6%)	4/51 (8%)	2 / 4
BROWSER_TRACE	1/48 (2%)	0/51 (0%)	1 / 0
JUDGE_RATIONALE	0/48 (0%)	2/51 (4%)	0 / 1
HUMAN_ANNOTATIONS	0/48 (0%)	1/51 (2%)	0 / 1
FINAL_STATE_SNAPSHOT	2/48 (4%)	1/51 (2%)	1 / 1
NONE	21/48 (44%)	16/51 (31%)	12 / 8
NOT_REPORTED	15/48 (31%)	20/51 (39%)	9 / 14
OTHER	2/48 (4%)	4/51 (8%)	1 / 2
Components without a determinable value	33 of 81	15 of 66	

C Audit and evidence status

Table A8. AI evidence audit, first part: the auditor checks the coder's cited evidence (VERIFY rows)

Variable	Rows	Supported	Partly	Wrong value	Unsupported	Share fully supported
TS1	36	30	6	0	0	83%
TS7	50	42	6	2	0	84%
TS8	5	5	0	0	0	100%
ALL	91	77	12	2	0	85%

Table A9. AI evidence audit, second part: the auditor recodes from the source without seeing the coder's value (BLIND rows; exact match of value sets; not intercoder reliability)

Variable	Rows	Exact match	Share	Mean Jaccard
EV1	16	12	75%	0.88
ID7	17	17	100%	1.0
OBS1	13	12	92%	0.949
TS1	11	11	100%	1.0
TS2	22	20	91%	0.909
TS7	60	60	100%	1.0
VE1	8	3	38%	0.375
VE2	13	6	46%	0.615
ALL	160	141	88%	0.909

Table A10. Evidence status of the 4,185 analyzed codings (all variables)

Status	Count
directly_reported	2554
inferred	586
absent_checked	584
not_reported	269
not_applicable	173
calculated_from_reported	14
unverified	5