REALITYBENCH: SIMULATED ENVIRONMENTS TO EVALUATE A VISION LANGUAGE MODEL'S ABILITY TO DOCUMENT HUMAN PHYSICAL WORK

Anonymous authors
Paper under double-blind review

ABSTRACT

Augmenting worker capabilities in the physical world through guidance has been a decades-long goal for AI agents. One key supporting capability for agents is to be able to understand what the worker is doing, seeing through the human's camera or smart glasses, and assist them in documentation. Despite impressive advances in vision language models, their capabilities in these settings remain uncertain and are challenging to benchmark because most physical work data is proprietary, and involves too much variability to study in a principled manner. Unlike structured settings such as kitchens and warehouses that have predictable lighting, geometry, and objects, field-facing tasks such as rooftop solar installation and vehicle repair are unconstrained and high variance. Here, we model real-world physical work scenarios in Blender across eleven distinct environments, incorporating variants in both image acceptability and documentation. The acceptability task asks whether a particular view would be suitable proof-of-work and documentation asks whether an agent can understand the task-relevant state of the environment from that view. Across 26 vision-language models, we find that top models largely solve documentation when the answer is text in frame, but are weak at reading measurements and at judging small parts and spatial relations, such as whether a tape measure leans. Acceptability is harder to pin down because human raters disagree with each other. Measured against a scorer that agrees with the raters' majority about as often as a single rater does, the best models cannot be distinguished from a single non-author rater, at a cost within the range of a human pass. In the three environments where we can trace acceptability errors, most wrong rejections come from misreading the scene rather than the requirement. On the three documentation questions for which we have real images, model rankings on rendered frames transfer to the real images.

1 INTRODUCTION

It has been a long-term goal to create AI agents that augment human capabilities in the physical world, from early augmented-reality systems for manufacturing and maintenance (Caudell & Mizell, 1992; Feiner et al., 1993; Henderson & Feiner, 2011) to recent egocentric assistants (Wang et al., 2023b; Grauman et al., 2024; Huang et al., 2025). These agents can help novice human workers perform tasks that require knowledge, e.g., showing them where to find a particular part of a machine, or reduce documentation time for expert workers by converting a recording of their task performance into a structured report. We expect these capabilities to be of increasing interest.

Towards this goal, we evaluate performance of current vision language models (VLMs) on two kinds of tasks (Figure 1): *acceptability* and *documentation*. Acceptability tasks ask, given a natural language specification, whether an image satisfies it, for the purposes of documenting the state of the environment (e.g., to show that work was performed). For example, in solar installation, an installer may request a sharp, close-up picture that shows the flashing and lag bolt for every L-foot on a solar module mounting rail (glossary: Appendix B). A verdict must intersect several conditions at once, some of them geometric (is the tape seated flush? is the gauge on the module?). Documentation tasks ask, given a natural language specification, to extract certain information from the environment that

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.



Figure 1: The eleven environments of RealityBench with one scorer-acceptable frame from each. Because each scene is rendered in Blender, each frame can be scored programmatically. Under each frame: a short form of the environment's documentation question and the ground truth.

is needed to document the environment—for example, reading the height of a solar module above the roof off of a tape measure.

Evaluating the capability of agents on these tasks is challenging and expensive. In-field trials give the best ecological validity, but they are resource intensive. Archived in-field data exists in high volume, but suffers from two issues. First, it is restricted in access, limiting reproducibility and transparency. Second, it is imprecisely overseen: a single task may generate a huge volume of data (e.g., thousands of photos) with a human providing only a single label for an entire unit (*packet*), under business pressure, and often assisted by VLMs. The weakness of VLMs at certain visual tasks makes VLM-based processing of archived field data problematic. This creates mechanistic uncertainty—it is unclear what a VLM *should* be able to conclude from each image.

We therefore take a simulation approach. RealityBench renders eleven environments in Blender (Figure 1), each with an associated acceptability and documentation task, from rooftop solar installation (seven) and vehicle repair (four). This overcomes both issues—we release our environments publicly under a permissive license and task success is programmatically evaluable using the environment state. This programmatic evaluation is challenging in the same way alignment to human preferences is—humans may have different views on how the natural language specification should be translated into requirements. Nonetheless, on held-out frames the scorer agrees with the majority of our human raters on 82% of frames, close to the 76–92% at which an individual rater agrees with the others (Section 4). We evaluate 26 models on a benchmark of 1,378 frames from our simulator.

While simulation-based benchmarking has impressive flexibility, it is often difficult to know whether the results translate to the real world. We are able to estimate this gap through evaluations on in-field images procured from real-world physical work tasks, and real world reenactments (for one measuring task, which we release publicly). Model rankings transfer on all three matched documentation questions: across 22 models, Spearman $\rho \geq 0.63$ per question at $p \leq 0.002$. This is evidence that the simulated tasks predict which models do well on real photographs.

Our results provide insights into both what VLMs can perceive and how they judge. Documentation scores are high when the answer is text in frame (labels, plates, directory cards), except for long or unusual strings, and low on measurement (reading a tape at the crossing) and on judgments about small parts or relations (whether a tape leans, whether a lag bolt is present). On acceptability, the best models score about as well as a single non-author rater (F1 0.61 against 0.59). We trace acceptability errors in three environments where the acceptability and documentation tasks share a field. There, models apply the requirement consistently to what they report seeing, but they misread the scene often. For example, they judge 75% of upright tape measures as tilted, and these misreadings account

for most of their wrong rejections. For seven of the nine models we re-asked, the misreadings do not change when the verdict is asked last or not at all.

Contributions.

1. A set of eleven environments from two kinds of real-world physical work, each with a human-validated programmatic acceptability scorer for a particular natural language task and a programmatic documentation scorer. Since the environments are simulators, not image sets, others can extend them to new tasks and to closed-loop guidance.
2. An evaluation of 26 vision-language models on 1,378 frames from those environments, reporting acceptability against the scorer, documentation accuracy against the scene's ground truth, the cost of the whole set at list prices, and a comparison with a single human rater.
3. A sim-to-real comparison on three matched documentation questions using real photographs (a public tape-measure set that we release, and two proprietary sets of vehicle photographs), showing that rankings on rendered frames transfer to real photos.
4. An analysis of what makes the tasks hard for models, showing that in the three environments where we can trace them, most wrong rejections come from misreading the scene, not the natural language requirement.

2 Related Work

Assistants and datasets for physical work. Large egocentric corpora have the model observe the worker. Ego4D and Ego-Exo4D (Grauman et al., 2022; 2024) and procedural datasets such as Assembly101, IndustReal and CaptainCook4D (Sener et al., 2022; Schoonbeek et al., 2024; Peddi et al., 2024) label steps, mistakes (Lee et al., 2024) and proficiency. HoloAssist (Wang et al., 2023b) records a remote instructor guiding a worker in real time, and portable egocentric VLM assistants are beginning to appear (Huang et al., 2025). For inspections, VLMs have been evaluated as construction-safety inspectors (Chen & Zou, 2025) and as unified industrial-inspection models, where they proved brittle against task-specific baselines (Wallace et al., 2025). These datasets are video-centric and label the work—they don't ask whether a captured view is adequate proof that the work was done. We found no public dataset of worker-submitted proof-of-work photos judged for acceptability.

Synthetic environments and sim-to-real. Rendered data with exact ground truth has a long history as training data (Richter et al., 2016; Roberts et al., 2021) and as a diagnostic test bed, e.g., CLEVR (Johnson et al., 2017) and Super-CLEVR (Li et al., 2023), which varies scene factors independently. Generators include Kubric, BlenderProc and Infinigen (Greff et al., 2022; Denninger et al., 2023; Raistrick et al., 2023), and embodied simulators include ProcTHOR and Habitat (Deitke et al., 2022; Savva et al., 2019). Domain randomization (Tobin et al., 2017) and synthetic spatial training that improves real-image benchmarks (Ray et al., 2025) show that variation can improve sim-to-real transfer. We know of no prior work that measures whether VLM *rankings* on a rendered benchmark match rankings on real photos of the same task.

Judging image quality. No-reference image quality assessment scores perceptual quality from statistics (Mittal et al., 2012; 2013) or learned models (Ke et al., 2021), and recent work puts the question to VLMs directly: CLIP-IQA (Wang et al., 2023a) with antonym prompts, Q-Bench (Wu et al., 2024a) as a low-level-vision benchmark, and Q-Align and DepictQA (Wu et al., 2024b; You et al., 2024) for calibrated scores and language explanations. These judge quality independent of content. Task-defined acceptability appears in mobile document capture, where a frame is good if downstream OCR succeeds (Burie et al., 2015; Li et al., 2018; Bulatov et al., 2020), and in insurance imaging, where customer photos feed damage estimation (Mallios et al., 2023) and VLMs are benchmarked on claims imagery (Lin et al., 2025), but there the photos are assumed adequate.

Scoring an image against a specification. TIFA and Davidsonian Scene Graph (Hu et al., 2023; Cho et al., 2024) decompose a prompt into questions answered by a VQA model, VQAScore (Lin et al., 2024) asks a single yes/no, and GenEval (Ghosh et al., 2023) runs a hand-written detector pipeline over object state, reporting agreement with annotators close to inter-annotator agreement.

Visual programming (Gupta & Kembhavi, 2023; Surís et al., 2023; Cho et al., 2023) and LLM-written reward code over simulator state (Ma et al., 2024) generate the verifier itself. Our scorers are hand-written programs over privileged scene state, closest in spirit to GenEval, and we validate them against human ratings as that literature does. We do not use a VLM as the reference scorer, partly because studies of multimodal LLMs as judges find absolute scoring inconsistent even for frontier models (Chen et al., 2024b).

Fine-grained perception in VLMs. BLINK (Fu et al., 2024b) finds frontier models near chance on tasks humans solve at a glance, MMVP (Tong et al., 2024) traces failures on orientation, counting and small details to the CLIP encoder, and HallusionBench (Guan et al., 2024) separates answering from priors from misreading the image. Spatial benchmarks report that metric questions are among the hardest (Yang et al., 2025; Chen et al., 2024a), and PhysBench (Chow et al., 2025) covers physical properties and relations broadly. Reading text is well studied, e.g., through scene-text and document VQA (Singh et al., 2019; Mathew et al., 2021) and OCRBench (Liu et al., 2024; Fu et al., 2024a), which singles out non-semantic strings such as serial numbers as a weakness. Closest to our tape and gauge tasks, MeasureBench (Lin et al., 2026) benchmarks reading values off instruments in real and synthesized images and finds that frontier VLMs struggle, while specialist readers trained on synthetic gauges remain competitive (Leon-Alcazar et al., 2024; Mueez et al., 2026).

3 Environments

We construct 11 environments in Blender 4.5.12 in two settings: rooftop solar installation (seven) and vehicle repair (four). Each environment uses a shared model of the phone camera: a 36 mm-equivalent lens whose aperture is set per frame to match a drawn phone (sensor width 5.6–9.8 mm, f/1.6–2.4) so that Cycles renders a phone's depth of field. 70% of frames are focused on the target with a steady hand, 15% have the autofocus locked elsewhere and 15% have hand shake over a shutter time drawn log-uniformly from 1/500 to 1/25 s. Sensor noise, white balance, tone mapping, chromatic aberration, vignetting and a JPEG re-encode are applied after the render.

Each environment has an acceptability and documentation task. The acceptability tasks ask whether a photo satisfies a natural language condition. The documentation tasks ask the model to write some property of the environment to a structured JSON format. Example prompts are in Appendix C.

Why these tasks? First, they are derived from tasks where VLMs fell short on real field images. Second, they test models along many axes. For example, `rail_attachment` asks whether a close-up shows the flashing and lag bolt of a single roof attachment, testing recognition of small hardware in clutter. Other tasks ask the model to read text and capture an image that shows that text clearly to a human. Finally, while there is a body of VLM work in structured settings such as kitchens and warehouses that have predictable lighting, geometry, and objects, field-facing tasks such as rooftop solar installation and vehicle repair are unconstrained and high variance, and not rigorously studied. Together, these tasks allow us to assess perception and understanding in a nuanced way.

3.1 Rooftop Solar Installation

Six of the seven solar environments are on the tilted roof of a house where a solar array is being or has been installed. Five share one procedural roof scene: the roof rectangle and pitch, the shingle courses, the house body under the roof (lap siding or brick), a gable or shed roof, a chimney (about half of the scenes), a ladder leaning on the eave, the array layout and the sun are all drawn per scene from the generator's distributions. The background is a sky-only HDRI panorama with the sun lamp aligned to the panorama's own sun. `horizon_peak` uses one of 21 CC0 house meshes instead, re-covered with the same procedural shingle courses and trims, placed inside a full 360° panorama. The seventh, `breaker_label`, is set at the house's load center on a wall.

We follow an installation in order. The array's rails are held to the roof by L-feet, each lag-bolted into a rafter through a sheet of flashing slipped under the shingles which, together, anchor the rail and prevent leakage. `rail_attachment` asks for a close-up of one attachment that shows the L-foot, its flashing and the lag bolt, and asks whether the bolt and the flashing are present. Under the modules, the wire should not touch the shingles: `wire_adequacy` asks for a photograph of the wire's lowest point, and asks whether any wire touches the roof (the requirement also mentions

hard bends, which the scorer does not check separately). The module must stand clear of the roof: `tape_gap` asks for a tape measure standing up to the module frame, and asks for the number at the crossing, the true perpendicular gap and whether the tape leans. The module's slope is recorded with a digital angle gauge: `slope_gauge` asks for the gauge sitting on the module, aligned with the slope, with the module visible (the scorer also requires some of the module's edge in view, since alignment cannot be judged without it), and asks for the reading. Each piece of equipment is photographed by its nameplate: `equipment_label` asks for a legible nameplate and for its brand, model and serial number and what it is on. From the ridge, the installer photographs the skyline in every direction, which records the obstructions that shade the array. `horizon_peak` asks for a photo, taken from the peak, of the skyline in one direction with a little of the house's own roof along the bottom, and asks whether the horizon is visible and level. Finally, the array ties into the house's electrical panel: `breaker_label` asks for a square-on, legible close-up of the circuit directory card and for its full contents, the main breaker amperage and every circuit's name and amperage.

3.2 Vehicle Repair

The photographer is in a parking lot with multiple vehicles from a set of nine meshes, with the subject car identified by color and body style ("the silver sedan"). A damaged vehicle arriving at a repair shop is first documented as a whole. `car_view` asks for a particular face of the car (front, rear, driver or passenger side) straight on, with the whole car in frame from about two car lengths, and asks which face is shown and the body style. The damage is then photographed up close: in `damage_view`, a procedurally generated dent, scratch or rust mark on a panel must be photographed square-on from about an arm's length, with part of the adjacent panel in frame for context, and the model is asked the kind of damage and its severity (minor, moderate or severe, by surface area). The vehicle's identity is recorded: `car_plate` asks for the mounted license plate close up and legible, and for its characters and the state named on it. Finally the shop issues a repair order, written with the vehicle's VIN on a hand-held board and photographed with the car so the two are tied together: `ro_board` asks for the board, legible, with the car in the same frame, and for the RO number and the VIN.

4 Building the Benchmark Set and Acceptability Scorer

For each of the 11 scenes, we render 100 frames with the camera drawn from that environment's pose distribution. We perform additional positive sampling for the seven environments where random frames are rarely acceptable, for a total of 1,450 initial frames. In `ro_board`, 29 frames were withdrawn because of a defect in the board texture, leaving 1,421 frames.

We define an acceptability scorer for each environment, which is an intersection of acceptability terms. Some acceptability terms are computed directly from the scene after the camera is placed, including in-frame and occlusion by ray casting, distances, view angles, and pixels per unit length. Another set of terms is read off the rendered frame itself, including sharpness at the target, the text's contrast against its paper and its blown-out fraction, the wire's pixel count and edge contrast, and the label's paper luminance. Acceptability is orthogonal to work completeness—unacceptable work can be acceptably documented. Documentation answers are scored against the scene's ground truth. Details are in Tables 1 and 2. We call frames that pass the acceptability scorer *scorer-acceptable*.

We tune the scorer to simulate human judgment. Six raters (four authors and two colleagues) judged 30 random frames per environment for acceptability, against the requirement text the models see and without seeing the scorer's verdict: two raters judged all of them, three judged 20–25 per environment, and one author judged only `tape_gap` and `damage_view` (4.0–5.1 ratings per frame, Table 13).[1] We split this set once, by frame, into 60% used to adjust the scorer and set the gray zones (described below) and 40% looked at once, after the scorer was frozen. On each held-out frame with at least three ratings we take the raters' majority as the reference (95 frames after excluding ties). The tuned scorer agrees with that majority on 82% of frames (95% CI 75–90%). It errs on the strict side: it accepts 4 of the 62 frames the majority rejects and rejects 13 of the 33 that the majority accepts. An individual rater agrees with the majority of the *other* raters on 76–92% (67% for the author who rated two environments, on 18 frames, Table 13). We opted to drop the acceptability task for `car_plate`,

[1] Ten of the rated `ro_board` frames were among those later withdrawn; their ratings are kept in the scorer tuning and in the agreement figures below.

where human raters disagreed on small but legible plates, and the acceptability task for frames of `rail_attachment` that clearly show an attachment with its lag bolt or flashing missing, due to task ambiguity. We removed the scorer requirement that the `horizon_peak` horizon be level, which the requirement text did not ask for. After these reductions and the gray zones below, 1,378 frames remained. Of these, 1,265 are scored for acceptability, and the 547 scorer-acceptable frames (including those of `car_plate`) are scored for documentation.[2]

Acceptability gray zones. Natural language descriptions that are used in the field are ambiguous (what does "arm's length" mean?). We want to avoid marking down models that reasonably interpret natural language. Where the wording leaves a range open, or our human raters disagree with each other over it, we mark that range of the term as gray (four zones in Table 1). For example, in `damage_view`, which asks for a photo from about arm's length, camera distances between 1.5 and 3 m are gray. A frame whose verdict depends only on a gray term is not scored for any model. We drop 43 frames from the acceptability set this way.

Other metrics. We report the cost of the whole set at the developer's own list API price (Table 7), with thinking tokens priced at the output rate.[3] We served three models ourselves (Gemma 4 E4B, LFM2.5-VL-3B, Qwen3.5-4B), which we do not attempt to price due to differences between self-hosting and API costs. We report the mean number of thinking tokens per item (where a provider reports only an output count, the count minus the tokens of the reply we received) and the fraction of answers that match the stated grammar.

5 Real-World Data

For three of the tasks we have approximate real-world equivalents, which let us estimate the sim-to-real gap.

Reading tape measures. We find that reading the measured distance off a tape measure precisely is one of the most challenging synthetic tasks. To test whether this result transfers, we photographed a tape measure held against everyday objects (monitors, a laptop, benches, tables, a fence, a whiteboard, a car battery) and recorded the ground truth for each measurand with a caliper or a careful human tape read: 133 readings on 84 photos of 35 items across 12 scenes, which we release. Each reading is asked with a prompt of the same form as `tape_gap`'s, naming the object and the measurement in place of the module and the gap, and scored as read within $1/8$ in of the ground truth.

Assessing car damage. We use photographs of damaged vehicles from the real world. For `car_view`'s face question, one author labeled 117 real frames with every side and third of the car they show—the model's answer was marked correct if it names one of them. This is similar to `car_view`'s documentation task. For `damage_view`'s severity question, one author rated the 63 photos of damaged panels on the documentation task's severity scale.

6 Experiments

We evaluate the models shown in Table 5, including: 23 through their providers' APIs under provider default settings, and three small open-weight models (Gemma 4 E4B (Gemma Team, 2026), LFM2.5-VL-3B, Qwen3.5-4B) that we served ourselves in bf16 at the model-card recommended settings. We evaluate Gemma 4 E4B with thinking on and Qwen3.5-4B in its non-thinking mode: in thinking mode Qwen3.5-4B used a mean of 3,064 thinking tokens per item on a 180-item sample, about four times Gemma 4 E4B (740). We report the acceptability F1 and documentation score vs. cost to complete the entire set in Fig. 2. More experimental details, model performance statistics, and confidence intervals are in Appendix D. Four models were not run on the real photographs due to logistical issues: Qwen 3.8 Max, Muse Spark 1.3, Kimi K3 (Kimi Team, 2026), and DeepSeek V4.1 Flash (DeepSeek-AI, 2026).

[2] Tape leaning is scored on all 143 `tape_gap` frames as lean can be judged from all frames, and all scorer-acceptable frames have no lean.

[3] Cache benefits would be small due to the input tokens being dominated by the image. We report numbers without cache benefits.

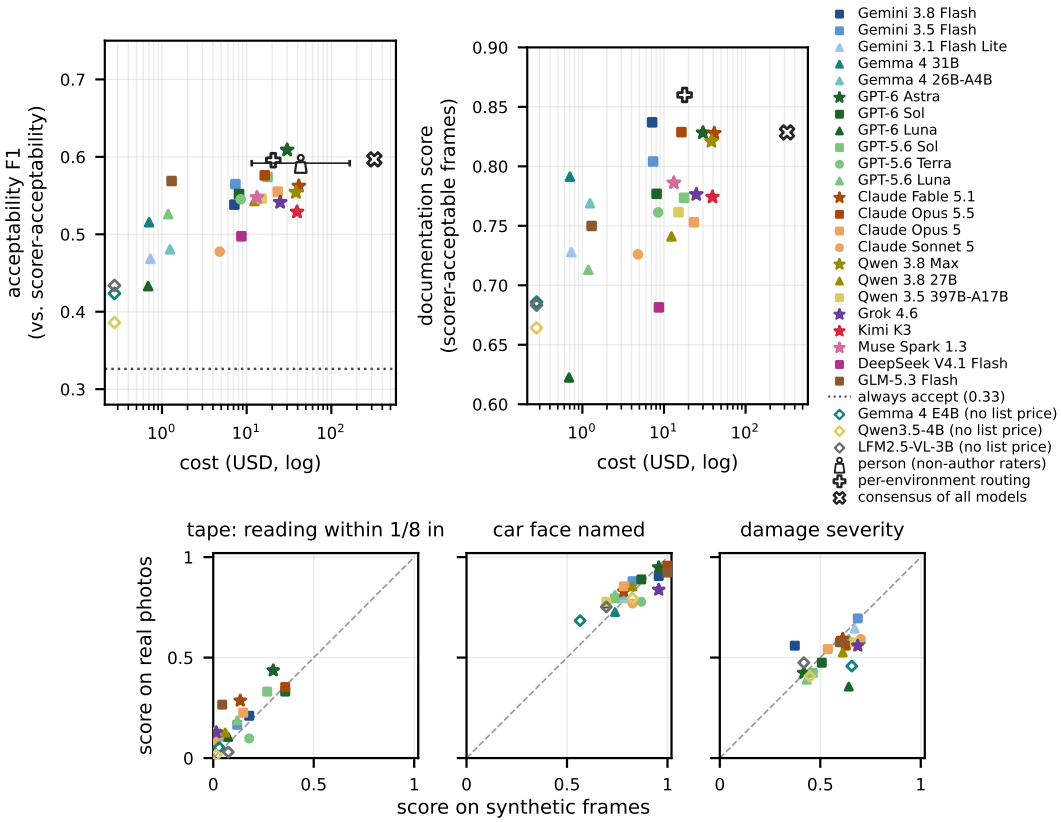

Figure 2: **Top:** acceptability F1 against scorer-acceptability (left) and documentation score on scorer-acceptable frames (right) against the cost of the whole set. Color marks the developer and shade the generation (darker is newer; Gemma 4 26B-A4B is a lighter shade only to tell it from Gemma 4 31B) and shape is a rough size tier (★ flagship, ■ large, ● mid, ▲ small, ♦ 3–4B). The dotted line is the "always accept" baseline; the hollow markers at the left edge are the small open-weight models we served ourselves. ⌖ routes each environment to the model that scored best on it (chosen on the other half of the frames) and ⊠ takes the majority verdict and answer of all models. 🧍 marks a single non-author rater's F1 (95% CI 0.50–0.67) at the cost of one human pass over the set ($11–165). **Bottom:** documentation score on synthetic frames (scorer-acceptable; for severity, frames where the damage is readable) against real photographs on the three matched questions.

6.1 Results

On synthetic data, models spread widely on both tracks (acceptability F1 0.39–0.61, documentation 0.62–0.84), with the two rankings in approximate agreement (Spearman $\rho = 0.69$, $p < 0.001$). On each track, we call the leader and the models whose paired 95% interval against it includes zero the *top group* (Table 6). On acceptability, GPT-6 Astra leads (0.61, $30.05), with Claude Opus 5.5 (0.58, $16.36), GLM-5.3 Flash (0.57, $1.30) and Gemini 3.5 Flash (0.56, $7.36) in the top group. On documentation, Gemini 3.8 Flash leads (0.84, $7.16), with Claude Opus 5.5 (0.83, $16.36), GPT-6 Astra (0.83, $30.05), Claude Fable 5.1 (0.83, $41.35) and Qwen 3.8 Max (0.82, $38.19) in the top group. GPT-6 Astra and Claude Opus 5.5 are the only models in both top groups. The best open-weight model on acceptability is GLM-5.3 Flash, and on documentation the much smaller Gemma 4 31B (0.79, $0.70). Routing across models helps documentation but not acceptability: routing each environment to the model that scored best on it (chosen on the other half of the frames) reaches a documentation score of 0.86 ($17.90), above every single model, but an F1 of only 0.60 ($20.60). The majority of all models reaches F1 0.60 and documentation 0.83 at $322—on par with or worse than the best single model.

On the cost axis, the models fall into three separated groups: those costing over $4 for the set ($4.80–41.35), six costing under $2 ($0.69–1.30, which we call moderately priced), and the unpriced models

we served ourselves. The moderately priced group contains three open-weight models (Gemma 4 31B, Gemma 4 26B-A4B and GLM-5.3 Flash) and three closed-weight models (GPT-6 Luna, Gemini 3.1 Flash Lite and GPT-5.6 Luna). The top group on acceptability within the moderately priced band is GLM-5.3 Flash (0.57, $1.30) and GPT-5.6 Luna (0.53, $1.18), and on documentation, Gemma 4 31B (0.79, $0.70) alone. Two of these three top group models are open-weight and thus make especially attractive targets for fine-tuning.

The unpriced band is on par with the lowest scoring priced models on both tracks, with LFM2.5-VL-3B and Gemma 4 E4B at the top of it (F1 0.43 and 0.42, documentation 0.68 and 0.69). These models are exciting because they can run on-device, with low latency and no network dependence.

Real-world images. Figure 2 (bottom) and Table 14 give, per model, the synthetic and real scores on the three matched questions. We see the tasks have broadly consistent difficulties between real and synthetic environments, with tape reading hardest on both, followed by damage severity, followed by naming the seen face of the car (which is saturated for top models). The rankings transfer, with Spearman ρ of 0.63–0.79 ($p \leq 0.002$) with the worst transfer on damage severity.[4]

Human raters. Using only the raters who were not involved in designing the scorer, a single rater scores an F1 of 0.59 (95% CI 0.50–0.67) against the scorer. Individual raters are noisy—the scorer tracks their majority. We cannot distinguish the strongest models from a single human rater on acceptability. We perform an approximate cost comparison. Our raters took 8–12 s per frame (including the documentation fields) while doing other work; they are not expert raters and did not know they were being timed, so we allow for faster work. At 3–12 s per item and $10/h (a somewhat arbitrary lower bound) to $35.91/h (U.S. Bureau of Labor Statistics, 2026), one human pass over the 1,378 items costs roughly $11–165, against $16.36 for Claude Opus 5.5 and $30.05 for GPT-6 Astra. Across generations, models improve on score and cost at once, so if the trend holds they will overtake a single rater on these tasks. Their remaining weaknesses are specific, and we turn to them next.

7 Analysis

What is easy and hard for models? Figure 3 divides the 24 documentation fields into three groups: *reading* (transcribe text or digits), *judgment* (a categorical scene state) and *measurement* (for reading tape measures) (details: Table 10). Models are very strong at reading: eight of the ten reading fields score 0.90–1.00 averaged over models. Long or unusual strings are still hard on average: the 17-character VIN (mean 0.59) and the gauge's seven-segment display (0.66), though Gemini 3.8 Flash scores 0.89 and 0.95. On judgment fields, those whose answer fills the frame are easy: what the label is on (mean 0.95), whether the horizon is visible (0.98) or level (0.91), the kind of damage (0.89), the car's face (0.83) and body style (0.81). The hard ones concern a small part or a relation: whether the lag bolt is present (0.67, best 0.82), how severe the damage is (0.61, best 0.75), whether the wire touches the roof (0.57, best 0.74) and whether the tape is leaning (0.55, best 0.82). Reading a tape measure is the hardest tested task: the number at the crossing is within $1/8$ in 11% of the time on average (best 36%, Opus 5.5 and GPT-6 Sol), and the true gap within $1/4$ in 23% (best 48%, Opus 5.5); as we saw, this difficulty carries over to real photographs. Across models, reading score is not detectably correlated with judgment (Spearman $\rho = 0.23$, $p = 0.27$), while judgment and measurement are strongly correlated ($\rho = 0.61$). Acceptability F1 follows judgment ($\rho = 0.77$) and measurement ($\rho = 0.77$), not reading ($\rho = 0.26$, $p = 0.19$).

Drivers of difficulty within an environment. In `tape_gap` the models read the tape's angle in the image rather than against the module. The tape is within 5° of upright on every scorer-acceptable frame, yet the models call it tilted 75% of the time, and the rate rises with the roll of the camera. The rise appears in 23 of the 26 models and in 5 of the 7 on the cost frontiers. Close shots with little roof for reference are the worst (Spearman -0.59 with roll, -0.51 with the apparent blade size). Reading fields are governed by resolution and viewing angle, as expected: the VIN on the RO board tracks pixels per text row ($+0.71$), the RO number the yaw of the board (-0.51), the breaker directory the viewing angle (-0.44). Two judgment fields are governed by visibility and vantage: whether the lag

[4] On `damage_view`, the real frames are not solely scorer-acceptable—thus, we use a sharpness-and-visibility-only condition to create the synthetic comparison set.

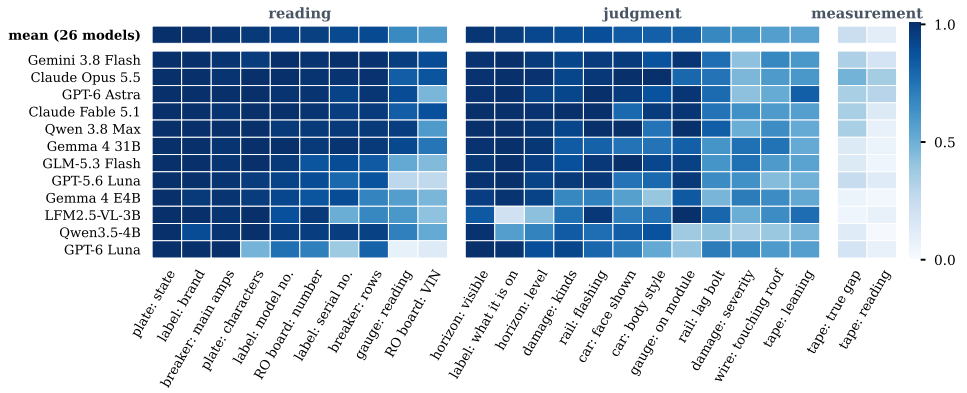

Figure 3: Documentation score by field (columns) and model (rows) on scorer-acceptable frames (tape leaning: all `tape_gap` frames). Columns are grouped by skill and sorted by the mean over all 26 models (top row). Rows are the models on a cost frontier of Figure 2, the other models in the documentation top group and the three models we served, sorted by documentation score.

bolt is present tracks how much of the mount is unoccluded ($+0.56$), and whether the gauge sits on the module tracks camera pitch ($+0.51$).

Do models misread the requirement or the scene? Acceptability F1 follows the geometric judgment fields, not reading. We analyze this through three environments where the acceptability and documentation tasks overlap: whether the tape is tilted, whether the gauge is on the module, whether the lag bolt and flashing are present. On scorer-acceptable frames the property always holds, so a model that reports the defect has misread the scene. Models apply the requirement to what they read—they reject 89–99.6% of the frames on which they report the defect—but they misread often, and the misreadings come with most of their wrong rejections: 97% in `tape_gap`, 74% in `slope_gauge` and 63% in `rail_attachment` (Table 12).

Is the misreading real, or a justification of the verdict? The reply gives the verdict first, so a model could reject a frame for another reason and then report a defect to justify it. We re-asked 150 frames from these environments to nine models with the verdict last and with no verdict at all (Appendix D). For seven, the defect reports do not change beyond run-to-run noise. The two cheapest, Gemini 3.1 Flash Lite and Gemma 4 31B, report fewer defects without the verdict (Flash Lite about half as many). For them, asking for the verdict first worsens their documentation answers.

Documentation on scorer-rejected frames. On the random frames, the documentation score is 0.71 on scorer-acceptable frames and 0.50 on the rest, and every model scores higher on scorer-acceptable frames (by 0.14–0.26). The gap is largest on the reading tasks (+0.58 to +0.70) and smallest in `horizon_peak` (+0.09), `car_view` (+0.07), `wire_adequacy` (+0.01) and `tape_gap` (-0.17). In `tape_gap` the tape leans on 88% of scorer-rejected frames, where the models' tendency to call the tape tilted is usually right.

8 Conclusion

We introduced RealityBench, eleven rendered environments from rooftop solar installation and vehicle repair. The environments are simulators, not image sets, so others can extend them to new tasks and to closed-loop guidance. Each environment poses two tasks a field agent needs, both scored programmatically from the scene: judging whether a photograph is adequate proof of work and reading the task-relevant state from it. Our observations show that reading text in frame is largely solved. The weaknesses are in reading measurements and in judging small parts and spatial relations. Acceptability, however, is harder to pin down, since raters disagree with each other, and the best models are level with a single non-author rater. In the three environments where we can trace acceptability errors, most wrong rejections come from misreading the scene. Rankings on rendered frames transfer to real photographs on all three matched documentation questions.

AI USE STATEMENT

In this work, we used generative AI tools for the following tasks with required disclosure: proposing or refining hypotheses; designing or providing feedback on research methodology or experiments; implementing methods; cleaning and reformatting data; and interpreting results. We have not used generative AI tools to generate synthetic data; assist with translation; or support qualitative and thematic data analysis. Helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs are not applicable to this work. Additionally, we used generative AI tools for tasks with recommended disclosure: creating and editing software code, creating or modifying scientific figures or images, drafting parts of a research paper, identifying and summarizing related literature, and editing a research paper to improve readability. We have reviewed all AI-assisted work. Every scorer threshold was set against human ratings collected without AI involvement, the model runs and their analysis scripts were checked by the authors, and AI-proposed interpretations were kept only where the authors could verify them. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Human ratings. The acceptability ratings used to validate the scorers and to compare models against people were produced by four of the authors and two colleagues, rating rendered frames of inanimate scenes; one author also labeled the vehicle photographs of Section 5 with the face shown and the damage severity. No personal data was collected from the raters, so no ethics review was sought.

Real photographs. The tape-measure photographs were taken by an author of their own objects; we release them. The vehicle damage photographs are used with permission. We use them only to score model answers, do not release them, do not reproduce them in the paper, and sent them only to providers whose data-retention terms met ours.

Conflicts of interest. The benchmark measures the general capability of third-party models rather than any product produced by the authors, and the API models were run through their public APIs at list prices and the three open-weight models on rented GPUs.

Assets and licenses. Every third-party 3D asset in the environments is CC0 or CC-BY and is listed with its license and attribution in the released manifest.

Potential harms. A system that judges photographs of completed work could be used to pressure workers or to automate rejection of legitimate work. We argue that better evaluation of model capabilities is important for making societal decisions about the use of AI in human work.

REPRODUCIBILITY STATEMENT

The environments, scorers and evaluation code will be publicly released and are available to the reviewers in the supplementary material, together with the exact rendered frames (with their scene state and pose, and the scorer verdict for each; the images exceed the upload limit and are at `https://osf.io/7jhnu/?view_only=db0b56045e3c488aaf4f2f11ded8ac69`), the rendering seeds and generator versions that produce them (up to render noise), the full prompts (Appendix C), the human ratings used for validation, and the raw per-model replies and token usage. Section 3 and Appendix B define each environment and its acceptability rule; Appendix D gives the sampling of the set, the test conditions, the scoring rules and tolerances, and the bootstrap procedure for every interval; Table 7 gives the prices and dates used for cost. The API models were queried through their public APIs between 13 and 23 September 2026 under provider default settings with one query per item, so re-running them will reproduce our numbers only up to provider nondeterminism and model updates. The three served models were run from their public Hugging Face weights in bf16 with the sampling settings of their model cards, and the serving script is part of the released code. Of the real photographs, the tape-measure set is released with its ground truth.

REFERENCES

Konstantin Bulatov, Daniil Matalov, and Vladimir V. Arlazarov. MIDV-2019: Challenges of the modern mobile-based document OCR. In *Twelfth International Conference on Machine Vision (ICMV 2019)*, 2020.

Jean-Christophe Burie, Joseph Chazalon, Mickaël Coustaty, Sébastien Eskenazi, Muhammad Muzzamil Luqman, Maroua Mehri, Nibal Nayef, Jean-Marc Ogier, Sophea Prum, and Marçal Rusiñol. ICDAR2015 competition on smartphone document capture and OCR (SmartDoc). In *13th International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1161–1165, 2015.

T. P. Caudell and D. W. Mizell. Augmented reality: An application of heads-up display technology to manual manufacturing processes. In *Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences*, 1992. doi: 10.1109/HICSS.1992.183317.

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brian Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. SpatialVLM: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14455–14465, 2024a.

Dongping Chen, Ruoxi Chen, Shilin Zhang, Yinuo Liu, Yaochen Wang, Huichi Zhou, Qihui Zhang, Yao Wan, Pan Zhou, and Lichao Sun. MLLM-as-a-Judge: Assessing multimodal LLM-as-a-Judge with vision-language benchmark. In *International Conference on Machine Learning (ICML)*, 2024b.

Xuezheng Chen and Zhengbo Zou. Are large pre-trained vision language models effective construction safety inspectors? *arXiv preprint arXiv:2508.11011*, 2025.

Jaemin Cho, Abhay Zala, and Mohit Bansal. Visual programming for text-to-image generation and evaluation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Jaemin Cho, Yushi Hu, Roopal Garg, Peter Anderson, Ranjay Krishna, Jason Baldridge, Mohit Bansal, Jordi Pont-Tuset, and Su Wang. Davidsonian scene graph: Improving reliability in fine-grained evaluation for text-to-image generation. In *International Conference on Learning Representations (ICLR)*, 2024.

Wei Chow, Jiageng Mao, Boyi Li, Daniel Seita, Vitor Guizilini, and Yue Wang. PhysBench: Benchmarking and enhancing vision-language models for physical world understanding. In *International Conference on Learning Representations (ICLR)*, 2025.

DeepSeek-AI. DeepSeek-V4.1-Flash: Pushing the limits of KV cache compression. Technical report, https://huggingface.co/deepseek-ai/DeepSeek-V4.1-Flash/blob/main/DeepSeek_V41_Tech_Report.pdf, 2026.

Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. ProcTHOR: Large-scale embodied AI using procedural generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Maximilian Denninger, Dominik Winkelbauer, Martin Sundermeyer, Wout Boerdijk, Markus Knauer, Klaus H. Strobl, Matthias Humt, and Rudolph Triebel. BlenderProc2: A procedural pipeline for photorealistic rendering. *Journal of Open Source Software*, 8(82):4901, 2023.

Steven Feiner, Blair MacIntyre, and Dorée Seligmann. Knowledge-based augmented reality. *Communications of the ACM*, 36(7):53–62, 1993. doi: 10.1145/159544.159587.

Ling Fu, Zhebin Kuang, Jiajun Song, Mingxin Huang, Biao Yang, Yuzhe Li, Linghao Zhu, Qidi Luo, Xinyu Wang, Hao Lu, Zhang Li, Guozhi Tang, Bin Shan, Chunhui Lin, Qi Liu, Binghong Wu, Hao Feng, Hao Liu, Can Huang, Jingqun Tang, Wei Chen, Lianwen Jin, Yuliang Liu, and Xiang Bai. OCRBench v2: An improved benchmark for evaluating large multimodal models on visual text localization and reasoning. *arXiv preprint arXiv:2501.00321*, 2024a.

Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A. Smith, Wei-Chiu Ma, and Ranjay Krishna. BLINK: Multimodal large language models can see but not perceive. In *European Conference on Computer Vision (ECCV)*, 2024b.

Gemma Team. Gemma 4 technical report, 2026.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. GenEval: An object-focused framework for evaluating text-to-image alignment. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023.

Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Abrham Gebreselasie, Cristina Gonzalez, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jachym Kolar, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Ziwei Zhao, Yunyi Zhu, Pablo Arbelaez, David Crandall, Dima Damen, Giovanni Maria Farinella, Christian Fuegen, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, Eugene Byrne, Zach Chavis, Joya Chen, Feng Cheng, Fu-Jen Chu, Sean Crane, Avijit Dasgupta, Jing Dong, Maria Escobar, Cristhian Forigua, Abrham Gebreselasie, Sanjay Haresh, Jing Huang, Md Mohaiminul Islam, Suyog Jain, Rawal Khirodkar, Devansh Kukreja, Kevin J Liang, Jia-Wei Liu, Sagnik Majumder, Yongsen Mao, Miguel Martin, Effrosyni Mavroudi, Tushar Nagarajan, Francesco Ragusa, Santhosh Kumar Ramakrishnan, Luigi Seminara, Arjun Somayazulu, Yale Song, Shan Su, Zihui Xue, Edward Zhang, Jinxu Zhang, Angela Castillo, Changan Chen, Xinzhu Fu, Ryosuke Furuta, Cristina Gonzalez, Prince Gupta, Jiabo Hu, Yifei Huang, Yiming Huang, Weslie Khoo, Anush Kumar, Robert Kuo, Sach Lakhavani, Miao Liu, Mi Luo, Zhengyi Luo, Brighid Meredith, Austin Miller, Oluwatumininu Oguntola, Xiaqing Pan, Penny Peng, Shraman Pramanick, Merey Ramazanova, Fiona Ryan, Wei Shan, Kiran Somasundaram, Chenan Song, Audrey Southerland, Masatoshi Tateno, Huiyu Wang, Yuchen Wang, Takuma Yagi, Mingfei Yan, Xitong Yang, Zecheng Yu, Shengxin Cindy Zha, Chen Zhao, Ziwei Zhao, Zhifan Zhu, Jeff Zhuo, Pablo Arbelaez, Gedas Bertasius, David Crandall, Dima Damen, Jakob Engel, Giovanni Maria Farinella, Antonino Furnari, Bernard Ghanem, Judy Hoffman, C. V. Jawahar, Richard Newcombe, Hyun Soo Park, James M. Rehg, Yoichi Sato, Manolis Savva, Jianbo Shi, Mike Zheng Shou, and Michael Wray. Ego-exo4d: Understanding skilled human activity from first- and third-person perspectives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J. Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam Laradji, Hsueh-Ti (Derek) Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, Cengiz Oztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang, Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, Dinesh Manocha, and Tianyi Zhou. HallusionBench: An

advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14375–14385, 2024.

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.

Steven Henderson and Steven Feiner. Exploring the benefits of augmented reality documentation for maintenance and repair. *IEEE Transactions on Visualization and Computer Graphics*, 17(10): 1355–1368, 2011. doi: 10.1109/TVCG.2010.245.

Yushi Hu, Benlin Liu, Jungo Kasai, Yizhong Wang, Mari Ostendorf, Ranjay Krishna, and Noah A. Smith. TIFA: Accurate and interpretable text-to-image faithfulness evaluation with question answering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Yifei Huang, Jilan Xu, Baoqi Pei, Yuping He, Guo Chen, Mingfang Zhang, Lijin Yang, Zheng Nie, Jinyao Liu, Guoshun Fan, Dechen Lin, Fang Fang, Kunpeng Li, Chang Yuan, Xinyuan Chen, Yaohui Wang, Yali Wang, Yu Qiao, and Limin Wang. An egocentric vision-language model based portable real-time smart assistant. *arXiv preprint arXiv:2503.04250*, 2025.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2901–2910, 2017.

Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. MUSIQ: Multi-scale image quality transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Kimi Team. Kimi k3: Open frontier intelligence, 2026.

Shih-Po Lee, Zijia Lu, Zekun Zhang, Minh Hoai, and Ehsan Elhamifar. Error detection in egocentric procedural task videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Juan Leon-Alcazar, Yazeed Alnumay, Cheng Zheng, Hassane Trigui, Sahejad Patel, and Bernard Ghanem. Learning to read analog gauges from synthetic data. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2024.

Hongyu Li, Fan Zhu, and Junhua Qiu. CG-DIQA: No-reference document image quality assessment based on character gradient. In *International Conference on Pattern Recognition (ICPR)*, 2018.

Zhuowan Li, Xingrui Wang, Elias Stengel-Eskin, Adam Kortylewski, Wufei Ma, Benjamin Van Durme, and Alan Yuille. Super-CLEVR: A virtual benchmark to diagnose domain robustness in visual reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Chenwei Lin, Hanjia Lyu, Xian Xu, and Jiebo Luo. INS-MMBench: A comprehensive benchmark for evaluating LVLMs' performance in insurance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

Fenfen Lin, Yesheng Liu, Haiyu Xu, Yue Chen, Zheqi He, Mingxuan Zhao, Miguel Hu Chen, Jin-Ge Yao, and Xi Yang. Do vision-language models measure up? Benchmarking visual measurement reading with MeasureBench. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 38544–38553, 2026.

Zhiqiu Lin, Deepak Pathak, Baiqi Li, Jiayao Li, Xide Xia, Graham Neubig, Pengchuan Zhang, and Deva Ramanan. Evaluating text-to-visual generation with image-to-text generation. In *European Conference on Computer Vision (ECCV)*, 2024.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xucheng Yin, Cheng-lin Liu, Lianwen Jin, and Xiang Bai. OCRBench: On the hidden mystery of OCR in large multimodal models. *Science China Information Sciences*, 2024.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Dimitrios Mallios, Xiaofei Li, Niall McLaughlin, Jesus Martinez Del Rincon, Clare Galbraith, and Rory Garland. Vehicle damage severity estimation for insurance operations using in-the-wild mobile images. *IEEE Access*, 11:78644–78655, 2023.

Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. DocVQA: A dataset for VQA on document images. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021.

Anish Mittal, Anush Krishna Moorthy, and Alan Conrad Bovik. No-reference image quality assessment in the spatial domain. *IEEE Transactions on Image Processing*, 21(12):4695–4708, 2012.

Anish Mittal, Rajiv Soundararajan, and Alan C. Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal Processing Letters*, 20(3):209–212, 2013.

Abdul Mueez, Aaditya Baranwal, Junior Chaj-Mejia, Guneet Bhatia, Jason T. Voelker, and Shruti Vyas. Vision-language models for analog gauge reading: An empirical study of specialization, transfer and reliability. *arXiv preprint arXiv:2608.17723*, 2026.

Rohith Peddi, Shivvrat Arya, Bharath Challa, Likhitha Pallapothula, Akshay Vyas, Bhavya Gouripeddi, Jikai Wang, Qifan Zhang, Vasundhara Komaragiri, Eric Ragan, Nicholas Ruozzi, Yu Xiang, and Vibhav Gogate. Captaincook4d: A dataset for understanding errors in procedural activities. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024.

Alexander Raistrick, Lahav Lipson, Zeyu Ma, Lingjie Mei, Mingzhe Wang, Yiming Zuo, Karhan Kayan, Hongyu Wen, Beining Han, Yihan Wang, Alejandro Newell, Hei Law, Ankit Goyal, Kaiyu Yang, and Jia Deng. Infinite photorealistic worlds using procedural generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Arijit Ray, Jiafei Duan, Ellis Brown, Reuben Tan, Dina Bashkirova, Rose Hendrix, Kiana Ehsani, Aniruddha Kembhavi, Bryan A. Plummer, Ranjay Krishna, Kuo-Hao Zeng, and Kate Saenko. SAT: Dynamic spatial aptitude training for multimodal language models. In *Conference on Language Modeling (COLM)*, 2025.

Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *European Conference on Computer Vision (ECCV)*, 2016.

Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M. Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied AI research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Tim J. Schoonbeek, Tim Houben, Hans Onvlee, Peter H. N. de With, and Fons van der Sommen. Industreal: A dataset for procedure step recognition handling execution errors in egocentric videos in an industrial-like setting. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2024.

Fadime Sener, Dibyadip Chatterjee, Daniel Shelepov, Kun He, Dipika Singhania, Robert Wang, and Angela Yao. Assembly101: A large-scale multi-view video dataset for understanding procedural activities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards VQA models that can read. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

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.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. Eyes wide shut? Exploring the visual shortcomings of multimodal LLMs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9568–9578, 2024.

U.S. Bureau of Labor Statistics. Occupational Employment and Wage Statistics, May 2025: 47-4011 Construction and Building Inspectors. https://data.bls.gov/oesprofile/, 2026. National estimates; accessed 23 September 2026.

Conor Wallace, Isaac Corley, and Jonathan Lwowski. Inspectvlm: Unified in theory, unreliable in practice. In *ICCV 2025 Workshop on Vision-based Industrial Inspection (VISION)*, 2025.

Jianyi Wang, Kelvin C.K. Chan, and Chen Change Loy. Exploring CLIP for assessing the look and feel of images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023a.

Xin Wang, Taein Kwon, Mahdi Rad, Bowen Pan, Ishani Chakraborty, Sean Andrist, Dan Bohus, Ashley Feniello, Bugra Tekin, Felipe Vieira Frujeri, Neel Joshi, and Marc Pollefeys. Holoassist: an egocentric human interaction dataset for interactive ai assistants in the real world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023b.

Haoning Wu, Zicheng Zhang, Erli Zhang, Chaofeng Chen, Liang Liao, Annan Wang, Chunyi Li, Wenxiu Sun, Qiong Yan, Guangtao Zhai, and Weisi Lin. Q-Bench: A benchmark for general-purpose foundation models on low-level vision. In *International Conference on Learning Representations (ICLR)*, 2024a.

Haoning Wu, Zicheng Zhang, Weixia Zhang, Chaofeng Chen, Liang Liao, Chunyi Li, Yixuan Gao, Annan Wang, Erli Zhang, Wenxiu Sun, Qiong Yan, Xiongkuo Min, Guangtao Zhai, and Weisi Lin. Q-Align: Teaching LMMs for visual scoring via discrete text-defined levels. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024b.

Jihan Yang, Shusheng Yang, Anjali W. Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10632–10643, 2025.

Zhiyuan You, Zheyuan Li, Jinjin Gu, Zhenfei Yin, Tianfan Xue, and Chao Dong. Depicting beyond scores: Advancing image quality assessment through multi-modal language models. In *European Conference on Computer Vision (ECCV)*, 2024.

A Limitations

We evaluate single frames, whereas workers would naturally produce a video. The scorer is one reading of each requirement, validated against six raters on 30 frames per environment, and it errs on the strict side. The human comparison uses non-expert raters, and its cost is a rough range. The error tracing covers wrong rejections in the three environments where acceptability and documentation share a field. The real-world comparison covers three documentation questions and no acceptability

Table 1: Environment, task and scorer summaries. Every capture task additionally requires *sharp at target* (defocus and motion blur at the target ≤ 5 px; 6 px for `car_plate`, `car_view`, `ro_board`, `damage_view`). *Acceptable* is the fraction of the random field-pose frames of each environment that the scorer accepts. Documentation question is a summary of the task—not the prompt given to the model.

Environment	Scene	Documentation question	Acceptability terms	Acceptable
`rail_attachment`	roof	At this roof attachment, is a lag bolt present on the L-foot, and is flashing installed under it?	attachment in frame ($\geq$ 90% of it); attachment unoccluded; bolt location unoccluded ($\geq$ 50% of the spot where the lag bolt sits, or would sit, visible); fills the frame (bbox $\geq$ 2%); not a wide shot ($\leq$ 1.8 m); *not scored when the lag bolt or flashing is missing*	0.12
`wire_adequacy`	roof	Is any wire or cable touching the roof surface?	a lead wire in frame ($\geq$ 50% of it); that wire's lowest point visible; that wire unoccluded ($\geq$ 30% of it); wire visible to the eye: wire pixels $\geq$ 0.05% of the frame; wire stands out: edge contrast $\geq$ 0.03; close enough to judge contact (cam within 2.5 m of that wire's low point)	0.39
`tape_gap`	roof	What number is on the tape where the module frame crosses it, what is your estimate of the true perpendicular module-to-roof gap in inches, and is the tape leaning rather than standing flush?	crossing in frame; crossing unoccluded; close enough to read 1/16in ticks ($\leq$0.65 m); tape seated flush (tilt $\leq 5^\circ$); photographed from the front of the blade; *gray: a lean over 5° that shows as at most 5° in the image*	0.17
`slope_gauge`	roof	What does the angle gauge read, in degrees, and is it sitting on the module (glass or frame) or on bare roof?	display in frame; display unoccluded; digits legible: $\geq$ 14 px digit height; display readable angle (view angle $\leq 65^\circ$); gauge on the module, not bare roof; display not blown out ($\leq$ 30% saturated); gauge aligned with the slope (|yaw| $\leq 15^\circ$, else it under-reads); array context: module glass visible in frame; module edge in view to judge alignment ($\geq$ 300 px of frame edge visible); text stands out in the render (Weber contrast $\geq$ 0.2); *gray: |yaw| 15–45°*	0.25
`equipment_label`	roof	What does it say on the equipment label (brand, model number and serial number) and what kind of equipment is it on?	label fully in frame; label unoccluded ($\geq$ 95% of it); legible: $\geq$ 10 px per text row; barcode resolvable: $\geq$ 1 px per module; square-on enough to OCR (view angle $\leq 60^\circ$); label not blown out ($\leq$ 30% saturated); site context: some of the equipment/mounting in view; label lit: paper luminance $\geq$ 0.05; text stands out in the render (Weber contrast $\geq$ 0.3)	0.06
`horizon_peak`	roof	Is the horizon visible in this frame, and is it roughly level?	horizon in frame, in the middle 80% of the image; taken from the peak: the house's own roof hides $\leq$ 10% of the horizon arc; skyline inside the frame: sky above it and the surroundings below it across $\geq$ 65% of the width; level (roll $\leq 10^\circ$); own roof visible along the bottom of the frame ($\geq$ 4% of the bottom quarter): proof it was taken from the peak; phone at or above the ridge line (camera height above the ridge $\geq$ 0 m)	0.19
`breaker_label`	wall	What is the main breaker amperage and every circuit's name and amperage?	card in frame; card unoccluded; legible: $\geq$ 11 px per text row; square-on (view angle $\leq 45^\circ$); tight: card bbox $\geq$ 4% of frame; text stands out in the render (Weber contrast $\geq$ 0.225); card not blown out ($\leq$ 30% saturated)	0.05
`car_view`	car	Which side of the vehicle is this (front, rear, driver side, passenger side)? What is the vehicle's body style (sedan, SUV, etc.)?	whole car in frame; not three-quarter: langle off the nearest facel $\leq 30^\circ$; about two car lengths (1.5–3 lengths); car fills $\geq$ 3% of frame; *gray: 3–4 car lengths*	0.23
`damage_view`	car	What kind of damage (dent, scratch, rust) is this, and how severe is it?	damage in frame; damage unoccluded; square onto the panel (view angle $\leq 45^\circ$); an arm away (0.3–3.0 m); adjacent panel context ($\geq$ 10% of a neighboring panel in frame); resolved: $\geq$ 3 px per cm at the damage; damage visible ($\geq$ 60 contrasting px in frame); *gray: distance 1.5–3 m*	0.38
`car_plate`	car	What is the license plate number and the state named on it?	plate in frame; plate unoccluded; legible: $\geq$ 12 px per character; straight-on (view angle $\leq 45^\circ$); fills most of the frame (bbox $\geq$ 2%); text stands out in the render (Weber contrast $\geq$ 0.1); plate not blown out ($\leq$ 30% saturated); *acceptability not scored*	–
`ro_board`	car	What is written on the repair order board? What is the body style of the vehicle (sedan, SUV, etc.)?	board fully in frame; board unoccluded; legible: $\geq$ 9 px per text row; square-on (view angle $\leq 40^\circ$); vehicle in the same frame ($\geq$ 50% of the car in frame); board not blown out ($\leq$ 30% saturated); text stands out in the render (Weber contrast $\geq$ 0.3)	0.18

Table 2: Scoring primitives. The first group is computed in the scene after the camera is placed; the second reads the rendered frame (the target's pixels are located by an object-index mask rendered from the same camera).

Primitive	Definition
in-frame / unoccluded fraction	Sample points on the target's surface (vertices *and* face interiors—corner-only sampling under-reports thin objects); project each; cast a ray from the camera; the point is visible if the first hit is the target.
bbox fraction	Area of the target's projected bounding box within the frame.
view angle	Angle between the viewing ray to the target and the target's surface normal (0° = square-on).
px per unit	Pixels spanned by a unit length at the target (text row height, character cell, inch, centimeter): the legibility proxy.
clearance	Signed distance from a wire's surface (beveled mesh) to the roof plane; ≤ 0 is contact.
apparent reading	For a tape, the blade point that projects onto the module's front-bottom edge from the camera (§3).
skyline / own-roof occlusion	For `horizon_peak` frames: per-column test of the true skyline (panorama tree line or the house's own silhouette) against the frame; ray-cast occlusion by the house's own roof.
scene state	A property the scene was built with, read directly: whether the lag bolt and the flashing exist at the attachment.
blur at target	Circle of confusion from focus error, aperture and focal length, plus motion-blur displacement of the target between shutter poses (px); cross-checked by a Laplacian on the target crop.
text contrast / blown fraction	Inside the label mask: Weber contrast of the ink level against the paper level, $(\text{paper} - \text{ink})/\text{paper}$, and the fraction of saturated pixels.
paper luminance	The paper level of the label in linear luminance: a dark label is unreadable whatever its contrast.
target pixels / edge contrast	Pixels covered by the target (a wire, a module edge) and the mean luminance step across its boundary to the surrounding pixels.

task. The eleven environments come from two kinds of work. We test general-purpose models zero-shot at provider defaults, not specialist readers or detectors, and model versions and prices change quickly.

B GLOSSARY

module a solar panel. It is clamped to a pair of rails, and each rail is held to the roof by L-feet lag-bolted through flashing into the rafters.

flashing the metal sheet slipped under the shingles at each roof attachment so the lag-bolt hole does not leak; only its lower part is exposed.

L-foot / lag the bracket that holds the rail to the roof, and the lag bolt through it into the rafter.

horizon from the peak photos taken from the ridge in every compass direction so shading obstructions can be read from the skyline.

apparent reading the number on the tape where the camera sees the module edge cross it; differs from the geometric crossing by parallax.

repair order board a hand-held board on which the shop writes the repair order (RO) number and the vehicle's VIN, photographed together with the vehicle so the two are tied to each other.

C PROMPT EXAMPLES

Each item is one query with the frame as the image. Every per-frame prompt has the same frame, with the technician (a solar installer, an electrician or a vehicle-intake technician), the *requirement*, the *question* and the *keys* filled in per environment (Table 3):

```
You are reviewing a photo taken by <technician> to document
their work; the photo criteria will be used to prove the
work was carried out correctly.  The photo we need:
<requirement> First:  is this photo adequate - does it
satisfy the requirement as a reviewer would judge it?  Then
answer the documentation question.  <question> Answer with
exactly one JSON object and nothing else - no prose, no code
fence - with exactly these keys:  "adequate":  true or false
(a JSON boolean):  whether the photo is adequate; <keys,
each with its grammar in words>.  Use these values exactly
as written; a value in any other form is scored as wrong.
Answer every key even if you judge the photo inadequate:
give your best reading of what is visible.
```

For `horizon_peak`, whose requirement is set-level, the frame instead asks whether this one photo belongs in such a set. The full prompt for `tape_gap`:

```
You are reviewing a photo taken by a solar installer to
document their work; the photo criteria will be used to
prove the work was carried out correctly.  The photo we
need:  Tape measure hooked at the roof surface and standing
up to the module frame; the number where the frame crosses
the tape must be legible, and the tape should be seated
flush (perpendicular to the roof).  First:  is this photo
adequate - does it satisfy the requirement as a reviewer
would judge it?  Then answer the documentation question.
What number (inches) is on the tape where the module frame
crosses it, what is your estimate of the true perpendicular
module-to-roof gap in inches, and is the tape leaning rather
than standing flush?  Answer with exactly one JSON object
and nothing else - no prose, no code fence - with exactly
these keys:  "adequate":  true or false (a JSON boolean):
whether the photo is adequate; "reading_in":  a number (JSON
number, no units, no quotes); "gap_in":  a number (JSON
number, no units, no quotes); "tilted":  true or false (a
JSON boolean).  Use these values exactly as written; a value
in any other form is scored as wrong.  Answer every key even
if you judge the photo inadequate:  give your best reading
of what is visible.
```

Table 3: The per-environment parts of the prompt.

Environment	Requirement	Question	Keys
`rail_attachment`	The L-foot, its flashing and the lag bolt, close enough to judge the attachment. Reject wide array shots where the foot is a speck.	At this roof attachment, is a lag bolt present on the L-foot, and is flashing installed under it?	`lag_present`: bool; `flashing_present`: bool
`tape_gap`	Tape measure hooked at the roof surface and standing up to the module frame; the number where the frame crosses the tape must be legible, and the tape should be seated flush (perpendicular to the roof).	What number (inches) is on the tape where the module frame crosses it, what is your estimate of the true perpendicular module-to-roof gap in inches, and is the tape leaning rather than standing flush?	`reading_in`: number; `gap_in`: number; `tilted`: bool
`wire_adequacy`	Photograph the underside of each array close enough to judge the wiring: lead wires and trunk cable visible, so a reviewer can tell whether any wire touches the roof plane or has a hard bend. Reject photos that do not clearly show the wiring under the modules.	Is any wire or cable touching the roof surface (the shingles)?	`touching`: bool
`slope_gauge`	A digital angle gauge placed directly on the module glass or frame, aligned with the slope, showing its degree reading; the display must be legible with the module it sits on visible in the photo. Reject a gauge placed on bare roof instead of the module, and a shot where the display is unreadable or washed out.	What does the angle gauge read, in degrees, and is it sitting on the module (glass or frame) or on bare roof?	`reading_deg`: number; `on_module`: bool
`equipment_label`	Close-up of the equipment data label — on the back of an installed module, on a microinverter under the array, or on the inverter — with the model number, the serial number and its barcode all legible, square-on enough to read. Some of the equipment or mounting should be in view so it is clearly a label on site, not a stock image. Reject a shot where the small text is a blur, the barcode is cut off or washed out by glare.	Read the equipment data label: brand, model number and serial number, and say what the label is on.	`brand`: string; `model`: string; `serial`: string; `on`: module/microinverter/inverter
`horizon_peak`	A set of photos taken from the roof peak for the shading analysis: turning in place, each photo shows the skyline in that direction, with sky above it and the surroundings below it, and a little of this house's own roof along the bottom as proof of where it was taken. Together the photos should go most of the way around: four or more directions with no large gap. Reject photos taken from below the peak where the roof blocks the view, photos with no roof in them at all, and sets that leave a large gap.	Is the horizon visible in this frame and roughly level, and does this house's own roof hide part of the far view?	`horizon_visible`: bool; `level`: bool; `own_roof_blocks`: bool
`damage_view`	Square onto the damaged area from about an arm's length, with the whole damaged region and part of the adjacent undamaged panel in frame, so a reviewer can judge which panel, how far the damage extends and how severe it is. Reject an oblique or distant shot where the damage is a smudge, and a tight crop with no panel context.	What kind of damage is this, and how severe is it? Severity by the size of the damaged area: minor = under about 10 x 10 cm (100 cm2), moderate = 100 to 500 cm2 (up to about 22 x 22 cm), severe = larger than that.	`damage_types`: list of dent/scratch/rust; `severity`: minor/moderate/severe
`car_view`	Straight-on to the requested face of the vehicle (or perpendicular to its flank) from about two car lengths, full width or length of the car visible, both corners in frame. Reject three-quarter angles, frames cropping a corner, and any other car in the lot standing in for it.	Which side of the vehicle is this (front, rear, driver side, passenger side)? Also give the vehicle's body style.	`side`: front/rear/driver/passenger; `body_style`: one of 10 body styles
`car_plate`	Straight-on close-up of the mounted plate on the vehicle, filling most of the frame, characters legible end to end. Reject a wide shot of the whole car where the plate is a smudge, and the plate of any other car in the lot.	Read the license plate: its characters and the state named on it.	`plate`: string; `state`: string
`ro_board`	The repair-order board (RO number, VIN, date) readable in the same frame as the vehicle: close and square enough that the writing reads, with the car clearly visible behind or beside it. Reject a wide shot where the board text is a smudge, and a board close-up with no vehicle in it.	Read the repair-order board: the RO number and the VIN. Also give the body style of the vehicle in the frame.	`ro`: string; `vin`: string; `body_style`: one of 10 body styles
`breaker_label`	Tight on the label card or handwritten directory beside the breaker rows so every circuit name and its amperage read cleanly. Reject a full-panel shot where the label text is illegible, and a bare breaker column with no labels in frame.	Read the whole circuit directory card: the main breaker amperage and every circuit's name and amperage.	`main_amps`: number; `entries`: list of {name, amps}

D EVALUATION DETAILS

Generation. Frames that fail the render sanity check (camera inside a mesh, black or blown frame, a car mesh intersecting scenery) are excluded before sampling. Because the scorer accepts only

Table 4: Verdict-order ablation: 150 items from `tape_gap`, `slope_gauge` and `rail_attachment` (25 per environment from the random frames and 25 from the positive stratum), re-asked under four prompts: the benchmark's (*first*: the verdict is the first key), verdict *last*, documentation *alone* (no verdict asked) and the verdict alone. A *defect claim* is the documentation boolean in the direction the requirement rejects (tape tilted; gauge not on the module; lag bolt or flashing absent). Paired columns count items whose claim is present under *first* only / under the other prompt only, with the exact McNemar p; *rerun* compares *first* with the same prompt's record in the main run (the noise floor). *Rejects*: frames rejected under *first* and when the verdict is asked alone. ‡ on a cost frontier of Fig. 2. One sample per prompt; provider-default thinking.

	defect claims (of 150)			paired change in claims vs. *first*			rejects	
Model	first	last	alone	last	alone (p)	rerun	first	alone (p)
Gemini 3.8 Flash‡	82	85	83	4/7	7/8 (1.00)	8/5	96	92 (.60)
Gemini 3.5 Flash	94	97	88	7/10	12/6 (.24)	7/6	104	108 (.50)
Gemini 3.1 Flash Lite	125	103	66	27/5	60/1 (<.001)	10/6	125	133 (.13)
Gemma 4 31B‡	92	96	78	5/9	18/4 (.004)	8/7	101	107 (.36)
GPT-6 Astra‡	58	53	56	8/3	9/7 (.80)	9/6	76	73 (.65)
GPT-5.6 Sol	63	70	61	7/14	12/10 (.83)	14/11	81	97 (.002)
Claude Opus 5	100	98	95	8/6	15/10 (.42)	6/14	113	141 (<.001)
Claude Sonnet 5	119	118	110	8/7	18/9 (.12)	4/4	122	120 (.84)
GLM-5.3 Flash‡	82	74	76	19/11	17/11 (.34)	12/12	91	105 (.03)

5–39% of random field poses (Table 1), the miss rate of a model cannot be estimated per environment from 100 random frames alone. Thus, for the seven environments where random frames are rarely acceptable we add a *positive stratum* of 50 scorer-acceptable frames obtained by rejection sampling from the same pose distribution. In `ro_board`, 21 random and 8 positive-stratum frames were withdrawn after generation because of a defect in the board texture. The false-accept rate is measured on the random frames only.

Test conditions. Each frame is one query: the reviewer framing and the checklist requirement, then one JSON object whose first key is the acceptability verdict and whose remaining keys are the documentation answer in a stated grammar, answered whatever the verdict (the effect of this order is tested below). Acceptability is scored against the scorer's verdict; documentation against the ground truth per field (exact match after normalization, which for text upper-cases and keeps only letters and digits; for the breaker directory, the fraction of the card's rows matched by a reply row with the same name and amperage, in any order, each reply row used once, extra rows not penalized; tolerances of $1/8$ in, $1/4$ in and $0.2°$ for the tape reading, the gap and the gauge reading).

Does the order of the answer matter? We re-ask 150 items from `tape_gap`, `slope_gauge` and `rail_attachment` under three more prompts that differ from the benchmark's only as stated: the verdict as the *last* key, with the two instructions swapped to match; the documentation question *alone*, with the same framing and requirement; and the verdict alone. The benchmark's own prompt is re-run to give the run-to-run noise. We ran 4 of the 7 models on the cost frontiers of Fig. 2 and 5 others (Table 4). Gemini 3.1 Flash Lite reports fewer defects when the verdict is moved last, and about half as many, in all three environments, when no verdict is asked; Gemma 4 31B reports fewer missing lag bolts and flashings in `rail_attachment` when no verdict is asked. No other model changes beyond the re-run noise, and no model's documentation score on these items is lower without the verdict.

Detailed results. Table 5 gives, per model on the same 1,378 frames, the false-accept and miss rates, acceptability precision, recall and F1 with a 95% interval, the documentation score with its interval, the fraction of answers in grammar, the cost of the set and the mean thinking tokens per item. Table 6 gives the paired difference in F1 and in documentation score for every pair of models, and whether its interval excludes zero (every model saw the same items, so the bootstrap resamples the items once and scores both models on that resample). Table 7 lists the token prices with their sources and dates. Table 8 breaks acceptability F1 down by environment and Table 9 the false-accept rate; Table 10 gives the documentation score per field. False-accept and miss rates are against the scorer's verdict; the documentation score is the mean, over the frames the scorer accepts, of each frame's mean field score (fields as above; the breaker main amperage and rows count half each). Costs use the

Table 5: Detailed statistics. False-accept rate (random frames the scorer rejects that the model accepts) and miss rate (scorer-acceptable frames, random and positive strata, that the model rejects); acceptability precision, recall and F1 against the scorer (precision on the 934 random frames, recall including the positive strata), with a 95% interval on F1; the documentation score on scorer-acceptable frames (mean per-field score) with its 95% interval; the fraction of answers in grammar; the cost of the set at the price in Table 7 (thinking at the output rate); mean thinking tokens per item. Below the rule: Gemma 4 E4B, Qwen3.5-4B and LFM2.5-VL-3B, which we served ourselves; their developers list no per-token price, so they have no cost. They sit below the same rule in the per-environment tables and are in every comparison except the cost frontier, which needs a price. Intervals: stratified bootstrap over items, 1,000 resamples, the random frames and each positive stratum resampled separately. Replies cut by the output cap: Qwen 3.8 27B 6, Qwen 3.5 397B-A17B 10, Kimi K3 6, DeepSeek V4.1 Flash 71, GLM-5.3 Flash 14.

Model	FA (%)	miss (%)	P	R	F1 [95% CI]	doc. [95% CI]	gram. (%)	cost ($)	think
Gemini 3.8 Flash	11	38	0.48	0.62	0.54 [0.48, 0.59]	0.84 [0.82, 0.86]	100	7.16	1083
Gemini 3.5 Flash	13	34	0.49	0.66	0.56 [0.51, 0.62]	0.80 [0.78, 0.83]	97	7.36	1114
Gemini 3.1 Flash Lite	16	42	0.39	0.58	0.47 [0.41, 0.52]	0.73 [0.70, 0.75]	98	0.73	0
Gemma 4 31B	15	37	0.44	0.63	0.52 [0.46, 0.57]	0.79 [0.77, 0.81]	100	0.70	677
Gemma 4 26B-A4B	17	36	0.39	0.64	0.48 [0.42, 0.53]	0.77 [0.75, 0.79]	99	1.24	1325
GPT-6 Astra	12	32	0.55	0.68	0.61 [0.56, 0.66]	0.83 [0.81, 0.85]	100	30.05	165
GPT-6 Sol	13	39	0.50	0.61	0.55 [0.50, 0.60]	0.78 [0.75, 0.80]	100	8.13	320
GPT-6 Luna	9	59	0.46	0.41	0.43 [0.38, 0.48]	0.62 [0.60, 0.65]	100	0.69	751
GPT-5.6 Sol	16	31	0.49	0.69	0.57 [0.52, 0.62]	0.77 [0.75, 0.80]	100	17.65	368
GPT-5.6 Terra	17	30	0.45	0.70	0.55 [0.49, 0.60]	0.76 [0.74, 0.78]	100	8.49	282
GPT-5.6 Luna	16	38	0.46	0.62	0.53 [0.47, 0.58]	0.71 [0.69, 0.74]	100	1.18	482
Claude Fable 5.1	12	37	0.51	0.63	0.56 [0.51, 0.61]	0.83 [0.81, 0.85]	100	41.35	267
Claude Opus 5.5	14	33	0.51	0.67	0.58 [0.52, 0.62]	0.83 [0.80, 0.85]	100	16.36	263
Claude Opus 5	11	43	0.54	0.57	0.56 [0.50, 0.60]	0.75 [0.73, 0.78]	100	23.32	345
Claude Sonnet 5	12	49	0.45	0.51	0.48 [0.42, 0.53]	0.73 [0.70, 0.75]	100	4.80	19
Qwen 3.8 Max	23	22	0.43	0.78	0.55 [0.50, 0.60]	0.82 [0.80, 0.84]	100	38.19	4231
Qwen 3.8 27B	15	32	0.45	0.68	0.54 [0.48, 0.60]	0.74 [0.72, 0.76]	99	12.36	2763
Qwen 3.5 397B-A17B	11	39	0.49	0.61	0.55 [0.49, 0.60]	0.76 [0.74, 0.79]	99	15.03	2799
Grok 4.6	19	28	0.43	0.72	0.54 [0.49, 0.59]	0.78 [0.75, 0.80]	100	24.88	2634
Kimi K3	20	30	0.42	0.70	0.53 [0.48, 0.58]	0.77 [0.75, 0.80]	99	39.31	1582
Muse Spark 1.3	11	38	0.49	0.62	0.55 [0.49, 0.60]	0.79 [0.76, 0.81]	100	13.22	1825
DeepSeek V4.1 Flash	18	32	0.39	0.68	0.50 [0.44, 0.55]	0.68 [0.65, 0.71]	95	8.67	4903
GLM-5.3 Flash	14	32	0.49	0.68	0.57 [0.51, 0.62]	0.75 [0.72, 0.77]	99	1.30	1460
Gemma 4 E4B	33	30	0.30	0.70	0.42 [0.37, 0.47]	0.69 [0.66, 0.71]	98	–	740
Qwen3.5-4B	18	50	0.31	0.50	0.39 [0.33, 0.44]	0.66 [0.64, 0.69]	95	–	0
LFM2.5-VL-3B	40	21	0.30	0.79	0.43 [0.39, 0.48]	0.68 [0.66, 0.71]	99	–	0

prices of Table 7 (the list API price of the model's developer, read from the developer's own pricing page on the date shown, whichever host served our calls), times the tokens the API reported: every prompt token at the full input rate, thinking tokens at the output rate. Google lists a per-token price for Gemma 4 26B only, so Gemma 4 26B-A4B and Gemma 4 31B are both priced at that rate.

Table 6: Paired differences between models on the same 1,378 frames, acceptability F1 (top) and documentation score (bottom). Models are sorted by the metric (its value beside the name); a cell is the column model's value minus the row model's, both as printed, in hundredths. **Bold**: the paired 95% interval of the unrounded difference excludes zero; gray: it does not (160 of 325 pairs for F1, 76 for documentation). § no developer list price (served by us). Intervals: a stratified bootstrap that resamples the items once and scores both models on the same resample (2000 resamples).

F1		GPT-6 Astra	Claude Opus 5.5	GPT-5.6 Sol	GLM-5.3 Flash	Gemini 3.5 Flash	Claude Fable 5.1	Claude Opus 5	Qwen 3.8 Max	GPT-6 Sol	Muse Spark 1.3	Qwen 3.5 397B-A17B	GPT-5.6 Terra	Qwen 3.8 27B	Grok 4.6	Gemini 3.8 Flash	Kimi K3	GPT-5.6 Luna	Gemma 4 31B	DeepSeek V4.1 Flash	Gemma 4 26B-A4B	Claude Sonnet 5	Gemini 3.1 Flash Lite	LFM2.5-VL-3B§	GPT-6 Luna	Gemma 4 E4B§
GPT-6 Astra	0.61																									
Claude Opus 5.5	0.58	3																								
GPT-5.6 Sol	0.57	**4**	1																							
GLM-5.3 Flash	0.57	4	1	0																						
Gemini 3.5 Flash	0.56	5	2	1	1																					
Claude Fable 5.1	0.56	**5**	2	1	1	0																				
Claude Opus 5	0.56	**5**	2	1	1	0	0																			
Qwen 3.8 Max	0.55	**6**	3	2	2	1	1	1																		
GPT-6 Sol	0.55	**6**	3	2	2	1	1	1	0																	
Muse Spark 1.3	0.55	**6**	3	2	2	1	1	1	0	0																
Qwen 3.5 397B-A17B	0.55	**6**	3	2	2	1	1	1	0	0	0															
GPT-5.6 Terra	0.55	**6**	3	2	2	1	1	1	0	0	0	0														
Qwen 3.8 27B	0.54	**7**	4	3	3	2	2	2	1	1	1	1	1													
Grok 4.6	0.54	**7**	4	3	3	2	2	2	1	1	1	1	1	0												
Gemini 3.8 Flash	0.54	**7**	4	3	3	2	2	2	1	1	1	1	1	0	0											
Kimi K3	0.53	**8**	**5**	**4**	4	3	3	3	2	2	2	2	2	1	1	1										
GPT-5.6 Luna	0.53	**8**	**5**	**4**	4	3	3	3	2	2	2	2	2	1	1	1	0									
Gemma 4 31B	0.52	**9**	**6**	**5**	**5**	**4**	4	4	3	3	3	3	3	2	2	2	1	1								
DeepSeek V4.1 Flash	0.50	**11**	**8**	**7**	**7**	**6**	**6**	**6**	**5**	**5**	5	**5**	5	**4**	4	4	3	3	2							
Gemma 4 26B-A4B	0.48	**13**	**10**	**9**	**9**	**8**	**8**	**8**	**7**	**7**	**7**	**7**	**7**	**6**	**6**	**6**	**5**	5	4	2						
Claude Sonnet 5	0.48	**13**	**10**	**9**	**9**	**8**	**8**	**8**	**7**	**7**	**7**	**7**	**7**	**6**	**6**	**6**	**5**	5	4	2	0					
Gemini 3.1 Flash Lite	0.47	**14**	**11**	**10**	**10**	**9**	**9**	**9**	**8**	**8**	**8**	**8**	**8**	**7**	**7**	**7**	**6**	**6**	**5**	3	1	1				
LFM2.5-VL-3B§	0.43	**18**	**15**	**14**	**14**	**13**	**13**	**13**	**12**	**12**	**12**	**12**	**12**	**11**	**11**	**11**	**10**	**10**	**9**	**7**	5	5	4			
GPT-6 Luna	0.43	**18**	**15**	**14**	**14**	**13**	**13**	**13**	**12**	**12**	**12**	**12**	**12**	**11**	**11**	**11**	**10**	**10**	**9**	**7**	5	5	4	0		
Gemma 4 E4B§	0.42	**19**	**16**	**15**	**15**	**14**	**14**	**14**	**13**	**13**	**13**	**13**	**13**	**12**	**12**	**12**	**11**	**11**	**10**	**8**	**6**	**6**	5	1	1	
Qwen3.5-4B§	0.39	**22**	**19**	**18**	**18**	**17**	**17**	**17**	**16**	**16**	**16**	**16**	**16**	**15**	**15**	**15**	**14**	**14**	**13**	**11**	**9**	**9**	**8**	**4**	4	3

Documentation		Gemini 3.8 Flash	Claude Opus 5.5	GPT-6 Astra	Claude Fable 5.1	Qwen 3.8 Max	Gemini 3.5 Flash	Gemma 4 31B	Muse Spark 1.3	GPT-6 Sol	Grok 4.6	Kimi K3	GPT-5.6 Sol	Gemma 4 26B-A4B	Qwen 3.5 397B-A17B	GPT-5.6 Terra	Claude Opus 5	GLM-5.3 Flash	Qwen 3.8 27B	Gemini 3.1 Flash Lite	Claude Sonnet 5	GPT-5.6 Luna	Gemma 4 E4B§	LFM2.5-VL-3B§	DeepSeek V4.1 Flash	Qwen3.5-4B§
Gemini 3.8 Flash	0.84																									
Claude Opus 5.5	0.83	1																								
GPT-6 Astra	0.83	1	0																							
Claude Fable 5.1	0.83	1	0	0																						
Qwen 3.8 Max	0.82	2	1	1	1																					
Gemini 3.5 Flash	0.80	**4**	**3**	3	**3**	2																				
Gemma 4 31B	0.79	**5**	**4**	**4**	**4**	**3**	1																			
Muse Spark 1.3	0.79	**5**	**4**	**4**	**4**	**3**	1	0																		
GPT-6 Sol	0.78	**6**	**5**	**5**	**5**	**4**	**2**	1	1																	
Grok 4.6	0.78	**6**	**5**	**5**	**5**	**4**	**2**	1	1	0																
Kimi K3	0.77	**7**	**6**	**6**	**6**	**5**	**3**	2	2	1	1															
GPT-5.6 Sol	0.77	**7**	**6**	**6**	**6**	**5**	**3**	2	2	1	1	0														
Gemma 4 26B-A4B	0.77	**7**	**6**	**6**	**6**	**5**	**3**	**2**	2	1	1	0	0													
Qwen 3.5 397B-A17B	0.76	**8**	**7**	**7**	**7**	**6**	**4**	**3**	**3**	2	2	1	1	1												
GPT-5.6 Terra	0.76	**8**	**7**	**7**	**7**	**6**	**4**	**3**	**3**	2	2	1	1	1	0											
Claude Opus 5	0.75	**9**	**8**	**8**	**8**	**7**	**5**	**4**	**4**	3	3	2	2	2	1	1										
GLM-5.3 Flash	0.75	**9**	**8**	**8**	**8**	**7**	**5**	**4**	**4**	**3**	**3**	**2**	2	2	1	1	0									
Qwen 3.8 27B	0.74	**10**	**9**	**9**	**9**	**8**	**6**	**5**	**5**	**4**	**4**	**3**	**3**	**3**	2	2	1	1								
Gemini 3.1 Flash Lite	0.73	**11**	**10**	**10**	**10**	**9**	**7**	**6**	**6**	**5**	**5**	**4**	**4**	**4**	**3**	**3**	**2**	2	1							
Claude Sonnet 5	0.73	**11**	**10**	**10**	**10**	**9**	**7**	**6**	**6**	**5**	**5**	**4**	**4**	**4**	**3**	**3**	**2**	2	1	0						
GPT-5.6 Luna	0.71	**13**	**12**	**12**	**12**	**11**	**9**	**8**	**8**	**7**	**7**	**6**	**6**	**6**	**5**	**5**	**4**	**4**	**3**	2	2					
Gemma 4 E4B§	0.69	**15**	**14**	**14**	**14**	**13**	**11**	**10**	**10**	**9**	**9**	**8**	**8**	**8**	**7**	**7**	**6**	**6**	**5**	**4**	**4**	2				
LFM2.5-VL-3B§	0.68	**16**	**15**	**15**	**15**	**14**	**12**	**11**	**11**	**10**	**10**	**9**	**9**	**9**	**8**	**8**	**7**	**7**	**6**	**5**	**5**	3	1			
DeepSeek V4.1 Flash	0.68	**16**	**15**	**15**	**15**	**14**	**12**	**11**	**11**	**10**	**10**	**9**	**9**	**9**	**8**	**8**	**7**	**7**	**6**	**5**	**5**	**3**	1	0		
Qwen3.5-4B§	0.66	**18**	**17**	**17**	**17**	**16**	**14**	**13**	**13**	**12**	**12**	**11**	**11**	**11**	**10**	**10**	**9**	**9**	**8**	**7**	**7**	**5**	3	2	2	
GPT-6 Luna	0.62	**22**	**21**	**21**	**21**	**20**	**18**	**17**	**17**	**16**	**16**	**15**	**15**	**15**	**14**	**14**	**13**	**13**	**12**	**11**	**11**	**9**	**7**	**6**	**6**	**4**

Table 7: Prices behind the costs (USD per 1M tokens; thinking priced at the output rate), where the price was read and on which day. Every price is the list price of the model's developer on its own API, whichever host served our calls (Alibaba Cloud prices differ by deployment region; we use International, the only one outside China offered for all three Qwen models). DeepSeek is the only developer that prices by time of day, off-peak at half of peak; we quote the peak rate. † Google lists a per-token price for Gemma 4 26B only; we use it for Gemma 4 31B as well.

Model	input	output	source	read
Gemini 3.8 Flash	0.75	3.75	Google[a]	2026-09-13
Gemini 3.5 Flash	0.75	3.75	Google[a]	2026-09-13
Gemini 3.1 Flash Lite	0.30	2.50	Google[a]	2026-09-13
Gemma 4 31B†	0.15	0.60	Google Cloud (Vertex AI)[b]	2026-09-20
Gemma 4 26B-A4B	0.15	0.60	Google Cloud (Vertex AI)[b]	2026-09-20
GPT-6 Astra	10.00	50.00	OpenAI[c]	2026-09-15
GPT-6 Sol	2.00	10.00	OpenAI[c]	2026-09-23
GPT-6 Luna	0.10	0.50	OpenAI[c]	2026-09-23
GPT-5.6 Sol	4.00	20.00	OpenAI[c]	2026-09-15
GPT-5.6 Terra	2.00	12.00	OpenAI[c]	2026-09-15
GPT-5.6 Luna	0.20	1.20	OpenAI[c]	2026-09-15
Claude Fable 5.1	10.00	50.00	Anthropic[d]	2026-09-15
Claude Opus 5.5	4.00	20.00	Anthropic[d]	2026-09-23
Claude Opus 5	5.00	25.00	Anthropic[d]	2026-09-15
Claude Sonnet 5	2.00	10.00	Anthropic[d]	2026-09-15
Qwen 3.8 Max	2.00	6.00	Alibaba Cloud (International)[e]	2026-09-20
Qwen 3.8 27B	0.50	3.00	Alibaba Cloud (International)[e]	2026-09-20
Qwen 3.5 397B-A17B	0.60	3.60	Alibaba Cloud (International)[e]	2026-09-20
Grok 4.6	2.00	6.00	xAI[f]	2026-09-15
Kimi K3	3.00	15.00	Moonshot AI[g]	2026-09-20
Muse Spark 1.3	1.25	4.25	Meta[h]	2026-09-20
DeepSeek V4.1 Flash	0.30	1.20	DeepSeek[i]	2026-09-20
GLM-5.3 Flash	0.15	0.50	Z.AI[j]	2026-09-20
Gemma 4 E4B	–	–	no list price (served by us)	
Qwen3.5-4B	–	–	no list price (served by us)	
LFM2.5-VL-3B	–	–	no list price (served by us)	

[a] https://ai.google.dev/gemini-api/docs/pricing
[b] https://cloud.google.com/vertex-ai/generative-ai/pricing
[c] https://developers.openai.com/api/docs/pricing
[d] https://platform.claude.com/docs/en/about-claude/pricing
[e] https://www.alibabacloud.com/help/en/model-studio/model-pricing
[f] https://docs.x.ai/docs/models [g] https://platform.kimi.ai/docs/pricing/chat
[h] https://dev.meta.ai/docs/pricing-rate-limits
[i] https://api-docs.deepseek.com/quick_start/pricing [j] https://docs.z.ai/guides/overview/pricing

Table 8: Acceptability F1 per environment: positives are the frames the scorer accepts; precision on the random frames, recall on the random frames plus the positive stratum.

Model	breaker	plate	car view	damage	equip.	horizon	rail	RO board	slope	tape	wire
Gemini 3.8 Flash	0.45	–	0.38	0.14	0.41	0.54	0.45	0.73	0.62	0.40	0.24
Gemini 3.5 Flash	0.41	–	0.53	0.13	0.39	0.65	0.60	0.59	0.71	0.21	0.47
Gemini 3.1 Flash Lite	0.34	–	0.49	0.14	0.28	0.55	0.31	0.62	0.37	0.37	0.43
Gemma 4 31B	0.46	–	0.39	0.08	0.37	0.60	0.36	0.66	0.57	0.16	0.48
Gemma 4 26B-A4B	0.38	–	0.25	0.25	0.35	0.54	0.55	0.59	0.50	0.20	0.34
GPT-6 Astra	0.66	–	0.61	0.26	0.62	0.63	0.48	0.73	0.63	0.72	0.55
GPT-6 Sol	0.59	–	0.60	0.34	0.56	0.63	0.44	0.66	0.64	0.37	0.45
GPT-6 Luna	0.40	–	0.47	0.32	0.60	0.30	0.35	0.65	0.26	0.40	0.30
GPT-5.6 Sol	0.78	–	0.58	0.41	0.45	0.61	0.48	0.71	0.59	0.45	0.54
GPT-5.6 Terra	0.58	–	0.55	0.24	0.46	0.62	0.47	0.68	0.53	0.38	0.48
GPT-5.6 Luna	0.59	–	0.53	0.30	0.45	0.55	0.39	0.71	0.68	0.17	0.49
Claude Fable 5.1	0.59	–	0.65	0.38	0.55	0.66	0.24	0.71	0.67	0.28	0.35
Claude Opus 5.5	0.55	–	0.65	0.27	0.47	0.70	0.44	0.64	0.71	0.34	0.56
Claude Opus 5	0.54	–	0.60	0.38	0.57	0.67	0.32	0.73	0.49	0.37	0.48
Claude Sonnet 5	0.58	–	0.48	0.27	0.55	0.52	0.31	0.76	0.23	–	0.37
Qwen 3.8 Max	0.38	–	0.63	0.29	0.34	0.62	0.46	0.67	0.64	0.19	0.63
Qwen 3.8 27B	0.47	–	0.46	0.47	0.38	0.61	0.49	0.71	0.60	0.12	0.34
Qwen 3.5 397B-A17B	0.45	–	0.14	0.27	0.41	0.65	0.49	0.72	0.70	–	0.29
Grok 4.6	0.41	–	0.21	0.34	0.48	0.64	0.48	0.64	0.52	0.37	0.60
Kimi K3	0.52	–	0.50	0.28	0.42	0.63	0.41	0.71	0.60	0.00	0.57
Muse Spark 1.3	0.54	–	0.51	0.26	0.47	0.56	0.33	0.66	0.65	0.26	0.18
DeepSeek V4.1 Flash	0.30	–	0.41	0.40	0.34	0.60	0.46	0.67	0.51	0.18	0.09
GLM-5.3 Flash	0.64	–	0.51	0.46	0.38	0.62	0.32	0.73	0.63	0.37	0.49
Gemma 4 E4B	0.26	–	0.46	0.40	0.35	0.47	0.29	0.61	0.45	0.24	0.36
Qwen3.5-4B	0.40	–	0.23	0.30	0.38	0.43	0.18	0.65	0.33	–	0.34
LFM2.5-VL-3B	0.23	–	0.57	0.53	0.26	–	0.24	0.45	0.56	0.47	0.44

Table 9: False-accept rate (%): frames the scorer rejects that the model accepts, on the 100 random field-pose frames per environment.

Model	breaker	plate	car view	damage	equip.	horizon	rail	RO board	slope	tape	wire
scorer-acceptable (%)	5	21	26	28	6	19	12	18	27	18	39
Gemini 3.8 Flash	9	–	5	5	18	11	5	15	33	3	7
Gemini 3.5 Flash	15	–	8	6	20	12	3	29	20	1	11
Gemini 3.1 Flash Lite	20	–	12	3	32	21	11	26	8	4	20
Gemma 4 31B	9	–	16	2	21	22	3	22	33	3	20
Gemma 4 26B-A4B	17	–	8	18	23	26	8	29	33	1	8
GPT-6 Astra	0	–	17	15	5	15	7	8	41	5	23
GPT-6 Sol	0	–	16	39	10	19	0	11	36	4	10
GPT-6 Luna	1	–	9	11	6	20	5	6	12	8	10
GPT-5.6 Sol	0	–	17	24	13	16	3	12	50	12	21
GPT-5.6 Terra	6	–	14	21	15	21	8	14	55	5	18
GPT-5.6 Luna	2	–	30	24	13	19	14	11	20	3	33
Claude Fable 5.1	2	–	16	24	7	11	9	15	30	3	13
Claude Opus 5.5	7	–	19	23	11	12	19	12	9	4	25
Claude Opus 5	5	–	11	16	9	14	15	14	5	5	15
Claude Sonnet 5	6	–	14	13	10	22	25	12	5	0	8
Qwen 3.8 Max	17	–	22	27	24	21	18	20	42	0	48
Qwen 3.8 27B	12	–	11	19	20	22	8	15	28	0	15
Qwen 3.5 397B-A17B	13	–	5	3	18	16	2	15	18	0	15
Grok 4.6	15	–	3	24	14	22	7	23	50	4	34
Kimi K3	3	–	25	40	17	21	18	15	47	1	23
Muse Spark 1.3	6	–	9	5	12	17	3	14	38	1	2
DeepSeek V4.1 Flash	19	–	12	6	21	20	17	20	45	0	7
GLM-5.3 Flash	5	–	9	32	17	19	2	14	29	4	16
Gemma 4 E4B	29	–	53	52	22	35	27	28	61	17	18
Qwen3.5-4B	16	–	0	16	21	1	70	20	15	0	8
LFM2.5-VL-3B	36	–	39	50	37	0	69	52	59	49	5

Table 10: Documentation score by field and model on scorer-acceptable frames.

Field	Gemini 3.8 Flash	Gemini 3.5 Flash	Gemini 3.1 Flash Lite	Gemma 4 31B	Gemma 4 26B-A4B	Gemma 4 E4B	GPT-6 Astra	GPT-6 Sol	GPT-6 Luna	GPT-5.6 Sol	GPT-5.6 Terra	GPT-5.6 Luna	Claude Fable 5.1	Claude Opus 5.5	Claude Opus 5	Claude Sonnet 5	Qwen 3.8 Max	Qwen 3.8 27B	Qwen 3.5 397B-A17B	Qwen3.5-4B	Grok 4.6	Kimi K3	Muse Spark 1.3	DeepSeek V4.1 Flash	GLM-5.3 Flash	LFM2.5-VL-3B
breaker: main amperage	0.98	0.98	0.98	0.98	0.98	0.98	1.00	0.98	1.00	0.98	0.98	0.98	0.98	0.98	0.98	0.98	1.00	0.98	0.93	0.98	0.98	1.00	0.98	1.00	0.98	0.98
breaker: circuit rows (fraction right)	0.99	0.93	0.98	0.96	0.97	0.68	0.98	0.95	0.81	0.92	0.86	0.87	0.96	0.98	0.95	0.89	0.96	0.86	0.88	0.94	0.92	0.92	0.96	0.80	0.84	0.67
plate: characters	0.95	0.95	1.00	1.00	0.95	0.95	1.00	0.95	0.48	0.90	0.90	0.95	1.00	1.00	1.00	1.00	1.00	0.95	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
plate: state	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
label: brand	1.00	1.00	1.00	1.00	1.00	0.96	1.00	1.00	1.00	0.98	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.89	1.00	1.00	1.00	0.96	1.00	1.00
label: model number	1.00	0.96	1.00	1.00	0.98	0.91	0.98	0.84	0.75	0.89	0.79	0.93	1.00	1.00	0.98	0.96	0.96	0.93	1.00	0.95	0.98	0.98	0.98	0.95	0.96	0.88
label: serial number	1.00	0.95	1.00	0.98	1.00	0.89	0.93	0.93	0.38	0.89	0.93	0.79	0.95	0.98	1.00	0.93	0.96	0.96	0.91	0.95	0.96	1.00	0.96	0.93	0.89	0.50
RO board: RO number	0.98	0.98	0.98	0.96	0.98	0.86	0.98	0.88	0.70	0.89	0.95	0.89	0.98	0.98	0.98	0.98	0.96	0.88	0.93	0.95	0.96	0.98	0.96	0.96	0.86	0.96
RO board: VIN	0.89	0.82	0.80	0.73	0.73	0.46	0.46	0.32	0.12	0.39	0.43	0.27	0.88	0.86	0.88	0.86	0.59	0.45	0.61	0.52	0.61	0.73	0.59	0.43	0.45	0.41
gauge: display reading ($\pm 0.2°$)	0.95	0.95	0.95	0.91	0.88	0.57	0.89	0.55	0.08	0.57	0.41	0.28	0.83	0.83	0.53	0.63	0.95	0.51	0.81	0.69	0.52	0.59	0.64	0.41	0.52	0.60
car view: face shown	0.96	0.83	0.78	0.74	0.78	0.57	0.96	0.87	0.70	0.74	0.87	0.74	0.78	1.00	0.78	0.83	1.00	0.83	0.70	0.83	0.96	0.91	1.00	0.87	1.00	0.70
car view: body style	0.87	0.91	0.83	0.74	0.74	0.39	0.87	0.87	0.52	0.87	0.74	0.78	0.96	1.00	0.87	0.96	0.74	0.83	0.78	0.87	0.87	0.87	0.91	0.74	0.91	0.74
damage: kinds	0.92	0.92	0.83	0.83	0.83	0.67	0.92	0.92	0.92	0.92	0.96	0.96	1.00	0.92	0.96	0.88	0.92	0.88	0.96	0.83	0.92	0.92	0.83	0.92	0.88	0.75
damage: severity	0.42	0.67	0.67	0.75	0.71	0.71	0.42	0.58	0.67	0.58	0.67	0.62	0.62	0.46	0.62	0.71	0.50	0.67	0.62	0.33	0.62	0.67	0.67	0.75	0.75	0.50
label: what it is on	1.00	1.00	1.00	1.00	0.98	0.98	1.00	1.00	0.98	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.57	1.00	1.00	1.00	0.98	1.00	0.20
horizon: visible	0.99	0.99	0.91	0.99	0.99	0.94	0.99	1.00	1.00	0.99	1.00	0.99	1.00	0.99	0.99	0.99	1.00	1.00	1.00	0.99	0.97	1.00	0.99	0.91	0.97	0.84
horizon: level	0.96	0.96	0.94	0.97	0.80	0.93	0.93	0.99	0.88	0.91	0.96	0.93	0.99	0.99	0.97	0.94	0.97	0.94	0.90	0.68	0.96	0.97	0.97	0.88	0.94	0.42
rail: lag bolt present	0.76	0.61	0.50	0.61	0.68	0.47	0.77	0.61	0.71	0.77	0.76	0.65	0.74	0.74	0.60	0.61	0.82	0.68	0.69	0.40	0.73	0.73	0.53	0.71	0.61	0.79
rail: flashing present	0.95	0.74	0.37	0.81	0.84	0.69	1.00	1.00	0.79	1.00	0.97	0.97	1.00	0.97	0.81	0.92	1.00	0.89	0.85	0.76	0.97	0.89	0.94	0.94	0.97	0.97
gauge: on module vs bare roof	0.97	0.93	0.27	0.81	0.76	0.84	0.97	1.00	0.40	1.00	0.95	0.96	0.97	0.79	0.52	0.23	1.00	0.80	0.87	0.37	0.92	0.96	1.00	0.79	0.93	1.00
wire: touching roof	0.67	0.64	0.64	0.74	0.59	0.64	0.51	0.54	0.59	0.51	0.51	0.44	0.59	0.59	0.49	0.46	0.64	0.64	0.54	0.38	0.62	0.49	0.62	0.62	0.59	0.64
tape: leaning	0.60	0.54	0.57	0.52	0.52	0.52	0.82	0.59	0.57	0.62	0.60	0.48	0.57	0.60	0.59	0.48	0.52	0.48	0.48	0.47	0.59	0.45	0.57	0.29	0.55	0.76
tape: number at the crossing ($\pm 1/8$ in)	0.18	0.12	0.04	0.06	0.04	0.03	0.30	0.36	0.07	0.27	0.18	0.12	0.13	0.36	0.15	0.01	0.07	0.06	0.01	0.01	0.01	0.03	0.04	0.01	0.04	0.07
tape: true gap ($\pm 1/4$ in)	0.33	0.30	0.16	0.13	0.10	0.06	0.34	0.43	0.18	0.40	0.31	0.24	0.34	0.48	0.24	0.10	0.34	0.24	0.19	0.12	0.16	0.24	0.24	0.07	0.13	0.04

Table 11: Token price against set cost, models with a list price, sorted by output price. Prompt, visible output and thinking tokens are per item (means over the 1,378 items; Anthropic reports one output count, so thinking is that count minus the tokens of the reply we received, counted with the provider's tokenizer). *Prompt share* is the fraction of the set's cost that is prompt tokens; the rest is output and thinking at the output rate. The last two columns rank the models by output price and by the cost of the set.

Model	in $/M	out $/M	prompt	output	thinking	prompt share	set ($)	rank by price	rank by set
GLM-5.3 Flash	0.15	0.50	1,289	37	1,460	21%	1.30	1	6
GPT-6 Luna	0.10	0.50	1,020	41	751	20%	0.69	2	1
Gemma 4 26B-A4B	0.15	0.60	527	43	1,325	9%	1.24	3	5
Gemma 4 31B	0.15	0.60	527	43	677	15%	0.70	4	2
DeepSeek V4.1 Flash	0.30	1.20	786	143	4,903	4%	8.67	5	12
GPT-5.6 Luna	0.20	1.20	1,170	39	482	27%	1.18	6	4
Gemini 3.1 Flash Lite	0.30	2.50	1,333	52	0	75%	0.73	7	3
Qwen 3.8 27B	0.50	3.00	1,122	39	2,763	6%	12.36	8	13
Qwen 3.5 397B-A17B	0.60	3.60	1,027	60	2,799	6%	15.03	9	15
Gemini 3.5 Flash	0.75	3.75	1,333	43	1,114	19%	7.36	10	9
Gemini 3.8 Flash	0.75	3.75	1,333	36	1,083	19%	7.16	11	8
Muse Spark 1.3	1.25	4.25	1,221	74	1,825	16%	13.22	12	14
Grok 4.6	2.00	6.00	1,023	35	2,634	11%	24.88	13	19
Qwen 3.8 Max	2.00	6.00	1,040	41	4,231	8%	38.19	14	21
Claude Sonnet 5	2.00	10.00	1,406	48	19	81%	4.80	15	7
GPT-6 Sol	2.00	10.00	1,170	36	320	40%	8.13	16	10
GPT-5.6 Terra	2.00	12.00	1,170	37	282	38%	8.49	17	11
Kimi K3	3.00	15.00	1,370	46	1,582	14%	39.31	18	22
Claude Opus 5.5	4.00	20.00	1,408	50	263	47%	16.36	19	16
GPT-5.6 Sol	4.00	20.00	1,170	38	368	37%	17.65	20	17
Claude Opus 5	5.00	25.00	1,406	50	345	42%	23.32	21	18
Claude Fable 5.1	10.00	50.00	1,408	52	267	47%	41.35	22	23
GPT-6 Astra	10.00	50.00	1,170	37	165	54%	30.05	23	20

Table 12: Misreading and rejection on scorer-acceptable frames, where the property the requirement checks always holds, so the right verdict is accept and reporting the defect is a misreading. Pooled over the 26 models: model answers, how often the defect is reported, how often a report comes with a rejection in the same reply, how many frames are rejected, and what share of those rejections come with the defect reported. `rail_attachment` frames with a missing part are not scored for acceptability.

Environment	Defect reported	answers	reports defect	rejects if reported	rejected	of those, defect reported
`tape_gap`	tape is tilted	1,688	75%	99.6%	1,296	97%
`slope_gauge`	gauge is not on the module	1,950	19%	96.0%	488	74%
`rail_attachment`	lag bolt or flashing missing	1,273	35%	88.8%	626	63%

E HUMAN RATINGS

Table 13 gives, per environment, the random frames the raters judged, their agreement with each other, and the frozen scorer's agreement with their majority on the held-out frames (Section 4).

Table 13: Human acceptability ratings. *Frames*: random frames with at least two ratings (the frames used to adjust the scorer and the held-out frames); *agree*: raw agreement over all rater pairs on a frame; α: Krippendorff's alpha (nominal), which corrects for chance and allows raters to skip frames. *Scorer*: the frozen scorer's agreement with the raters' majority on the held-out frames with at least three ratings, ties excluded (all environments: 82% of 95). † acceptability not scored (the raters split on the requirement); frames the frozen scorer leaves unscored are not in the scorer column.

Set	Frames	Raters	Ratings/frame	Agree	α	Scorer (held-out)
`breaker_label`	30	5	4.1	0.94	0.79	0.91 (11)
`equipment_label`	29	5	4.3	0.90	0.58	0.91 (11)
`ro_board`	30	5	4.1	0.98	0.86	1.00 (8)
`rail_attachment`	30	5	4.1	0.88	0.53	0.80 (5)
`car_plate`†	28	5	4.2	0.80	0.54	–
`wire_adequacy`	30	5	4.3	0.78	0.56	0.91 (11)
`tape_gap`	30	6	5.1	0.72	0.44	0.73 (11)
`slope_gauge`	30	5	4.1	0.73	0.46	0.80 (10)
`car_view`	28	5	4.2	0.71	0.49	0.60 (10)
`horizon_peak`	30	5	4.0	0.65	0.25	0.89 (9)
`damage_view`	30	6	5.0	0.64	0.19	0.67 (9)

F REAL-PHOTO DETAILS

Table 14 gives the per-model numbers behind Figure 2 (bottom), with intervals.

Table 14: Synthetic against real, per model, three matched questions. Synthetic: the share of scorer-acceptable synthetic frames whose field is right (`tape_gap` reading within $1/8$ in; `car_view` face; `damage_view` severity on the frames where the scorer finds the damage readable: sharp, visible and resolved, framing not required). Real: our tape photos read within $1/8$ in (133 readings on 84 photos of 35 items); the face named is one the labels list as visible (117 side-determinable frames); severity on the 59 real damaged-panel photos an author rated on the prompt's area scale after the model runs, blind to the models' answers (4 rated can't tell are left out). Brackets: 95% bootstrap intervals over frames, or over items for the tape photos. Across the 22 models, synthetic against real (Spearman, Pearson): tape $\rho = 0.68$ ($p = 0.001$), $r = 0.82$ ($p < 0.001$); face $\rho = 0.79$ ($p < 0.001$), $r = 0.85$ ($p < 0.001$); severity $\rho = 0.63$ ($p = 0.002$), $r = 0.58$ ($p = 0.005$). Not run on the real photos: Qwen 3.8 Max, Muse Spark 1.3, Kimi K3, DeepSeek V4.1 Flash. Tape replies cut by the output cap, left out: GLM-5.3 Flash 54, Qwen 3.8 27B 15, Qwen 3.5 397B-A17B 8.

Model	tape: synthetic	tape: real	face: synthetic	face: real	severity: synthetic	severity: real
Claude Opus 5.5	0.36 [0.24, 0.48]	0.35 [0.26, 0.47]	1.00 [1.00, 1.00]	0.96 [0.91, 0.99]	0.63 [0.51, 0.73]	0.56 [0.44, 0.68]
GPT-6 Sol	0.36 [0.24, 0.48]	0.33 [0.25, 0.41]	0.87 [0.74, 1.00]	0.89 [0.83, 0.94]	0.51 [0.39, 0.63]	0.47 [0.36, 0.61]
GPT-6 Astra	0.30 [0.19, 0.40]	0.44 [0.34, 0.53]	0.96 [0.87, 1.00]	0.95 [0.91, 0.98]	0.42 [0.30, 0.54]	0.42 [0.31, 0.54]
GPT-5.6 Sol	0.27 [0.16, 0.37]	0.33 [0.25, 0.41]	0.74 [0.57, 0.91]	0.79 [0.72, 0.86]	0.46 [0.34, 0.58]	0.42 [0.31, 0.54]
Gemini 3.8 Flash	0.18 [0.09, 0.27]	0.21 [0.13, 0.29]	0.96 [0.87, 1.00]	0.91 [0.85, 0.96]	0.37 [0.27, 0.49]	0.56 [0.42, 0.68]
GPT-5.6 Terra	0.18 [0.09, 0.27]	0.10 [0.05, 0.16]	0.87 [0.70, 1.00]	0.78 [0.70, 0.85]	0.54 [0.42, 0.66]	0.54 [0.41, 0.68]
Claude Opus 5	0.15 [0.07, 0.24]	0.23 [0.15, 0.31]	0.78 [0.61, 0.91]	0.85 [0.79, 0.91]	0.54 [0.42, 0.66]	0.54 [0.41, 0.66]
Claude Fable 5.1	0.13 [0.06, 0.22]	0.29 [0.20, 0.37]	0.78 [0.61, 0.96]	0.83 [0.76, 0.90]	0.61 [0.49, 0.73]	0.59 [0.46, 0.71]
Gemini 3.5 Flash	0.12 [0.04, 0.21]	0.17 [0.10, 0.24]	0.83 [0.65, 0.96]	0.88 [0.82, 0.94]	0.69 [0.58, 0.79]	0.69 [0.58, 0.81]
GPT-5.6 Luna	0.12 [0.04, 0.19]	0.19 [0.13, 0.26]	0.74 [0.57, 0.91]	0.81 [0.74, 0.88]	0.43 [0.31, 0.54]	0.39 [0.27, 0.51]
GPT-6 Luna	0.07 [0.01, 0.13]	0.11 [0.05, 0.17]	0.70 [0.52, 0.87]	0.77 [0.69, 0.84]	0.64 [0.52, 0.76]	0.36 [0.24, 0.47]
LFM2.5-VL-3B	0.07 [0.01, 0.15]	0.03 [0.01, 0.06]	0.70 [0.52, 0.87]	0.75 [0.67, 0.83]	0.42 [0.30, 0.54]	0.47 [0.36, 0.59]
Gemma 4 31B	0.06 [0.01, 0.12]	0.11 [0.06, 0.16]	0.74 [0.57, 0.91]	0.73 [0.64, 0.80]	0.70 [0.60, 0.81]	0.59 [0.47, 0.71]
Qwen 3.8 27B	0.06 [0.01, 0.12]	0.13 [0.07, 0.19]	0.83 [0.65, 0.96]	0.85 [0.79, 0.91]	0.61 [0.51, 0.72]	0.53 [0.39, 0.66]
Gemini 3.1 Flash Lite	0.04 [0.00, 0.10]	0.11 [0.05, 0.17]	0.78 [0.61, 0.91]	0.79 [0.72, 0.86]	0.67 [0.55, 0.78]	0.64 [0.53, 0.76]
Gemma 4 26B-A4B	0.04 [0.00, 0.10]	0.11 [0.06, 0.19]	0.78 [0.61, 0.96]	0.81 [0.74, 0.88]	0.64 [0.52, 0.76]	0.59 [0.47, 0.71]
GLM-5.3 Flash	0.04 [0.00, 0.10]	0.27 [0.19, 0.35]	1.00 [1.00, 1.00]	0.92 [0.87, 0.97]	0.60 [0.48, 0.72]	0.58 [0.46, 0.71]
Gemma 4 E4B	0.03 [0.00, 0.07]	0.05 [0.02, 0.10]	0.57 [0.35, 0.74]	0.68 [0.59, 0.77]	0.66 [0.54, 0.76]	0.46 [0.32, 0.58]
Claude Sonnet 5	0.01 [0.00, 0.04]	0.08 [0.04, 0.14]	0.83 [0.65, 0.96]	0.77 [0.69, 0.84]	0.70 [0.58, 0.81]	0.59 [0.47, 0.71]
Grok 4.6	0.01 [0.00, 0.04]	0.13 [0.08, 0.19]	0.96 [0.87, 1.00]	0.84 [0.77, 0.90]	0.69 [0.57, 0.79]	0.56 [0.42, 0.68]
Qwen 3.5 397B-A17B	0.01 [0.00, 0.04]	0.13 [0.08, 0.18]	0.70 [0.52, 0.87]	0.78 [0.70, 0.85]	0.67 [0.55, 0.78]	0.58 [0.44, 0.71]
Qwen3.5-4B	0.01 [0.00, 0.04]	0.02 [0.00, 0.04]	0.83 [0.65, 0.96]	0.79 [0.72, 0.87]	0.45 [0.33, 0.57]	0.41 [0.29, 0.53]